CSEN 241 Cloud Computing HW1

Name: Li Huang
Student ID: 01641460

Environment

Computer	Dell
Model Name	Intel(R) Core(TM) i5-10400 CPU @ 2.90GHz
CPU	800 MHZ
Operating System	Linux fedora 6.6.12-200.fc39.x86_64

System Virtualization (QEMU) Setup

Installing QEMU on Linux(Fedora)

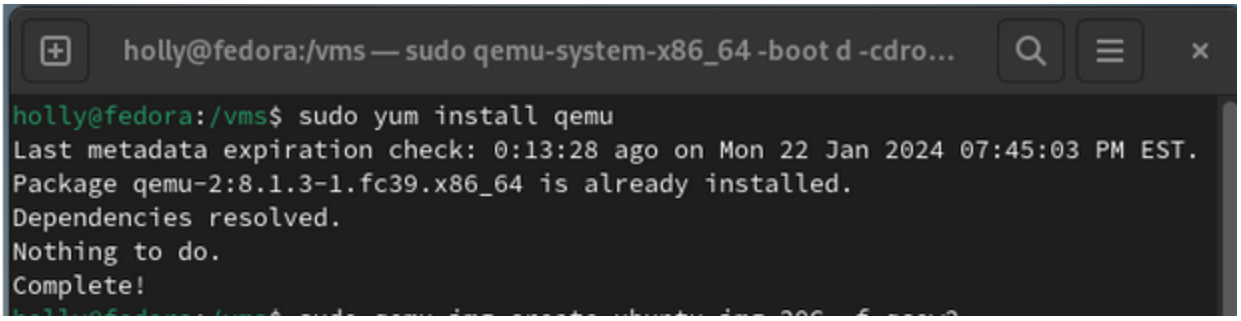
Download

download the x64 CPU Ubuntu 20.04 Server for Linux on the following link
https://releases.ubuntu.com/focal/

Install QEMU

From the terminal, run the following commands as a root user. Because Fedora does not use apt, I use yum instead.

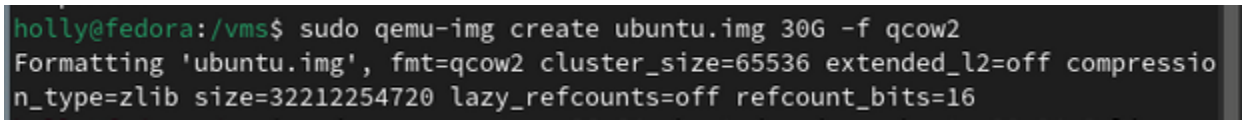
```
$ sudo yum install qemu
```



Create image

create the QEMU Image by running the following command.

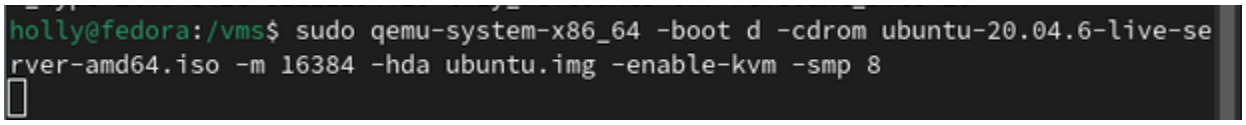
```
$ sudo qemu-img create ubuntu.img 10G -f qcow2
```



Install QEMU on the image

install the Ubuntu guest OS on the newly created QEMU image by running the following command:

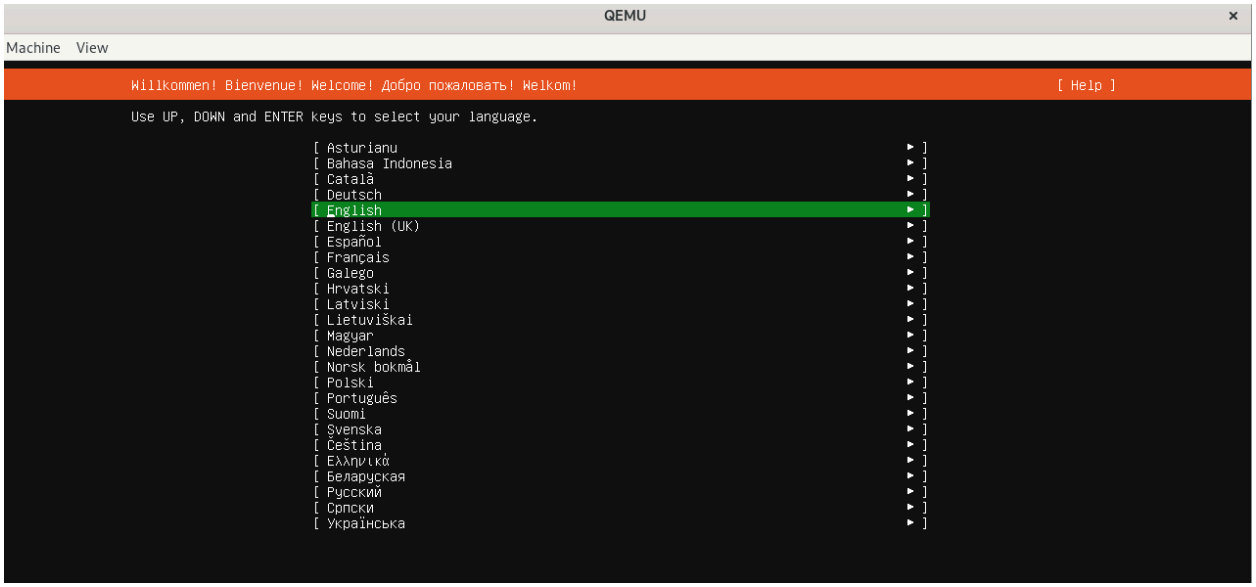
```
$  sudo qemu-system-x86_64 -boot d -cdrom ubuntu-20.04.6-live-server-amd64.iso -m 16384 -hda ubuntu.img -enable-kvm -smp 8
```

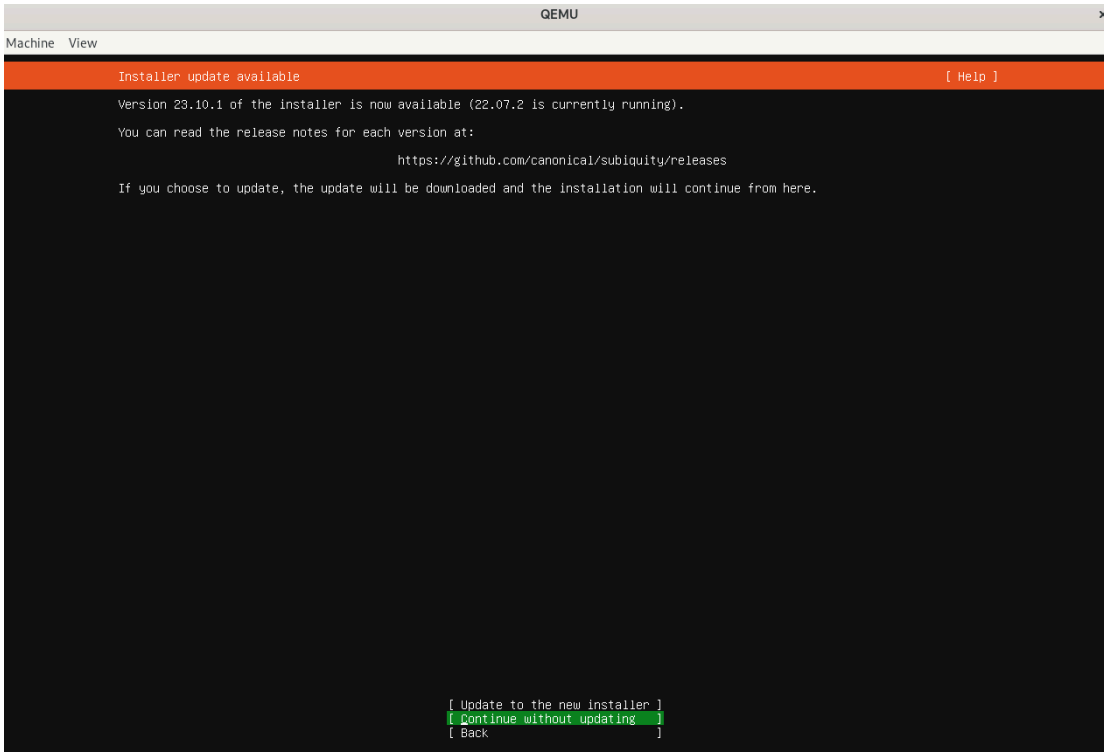


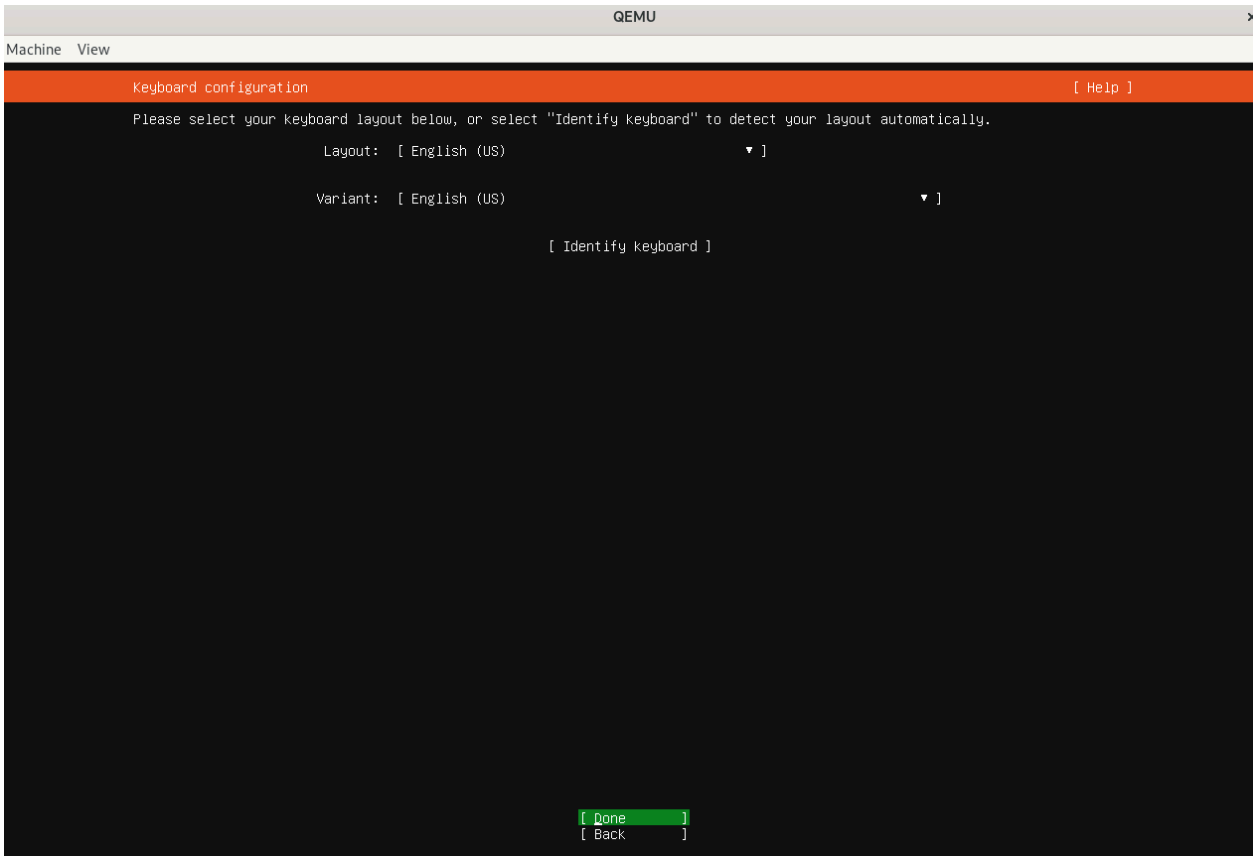
Follow the instructions

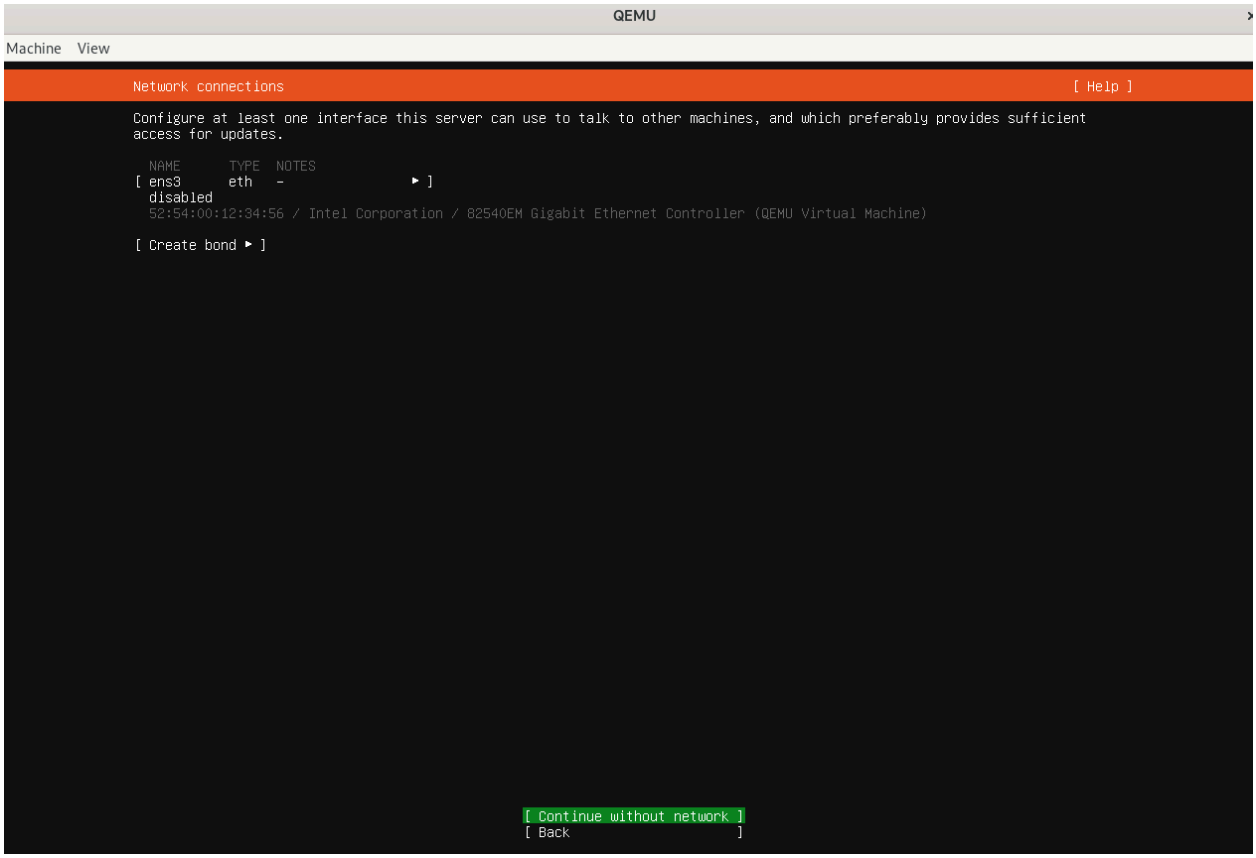
Then I just follow the instructions to finish the installation

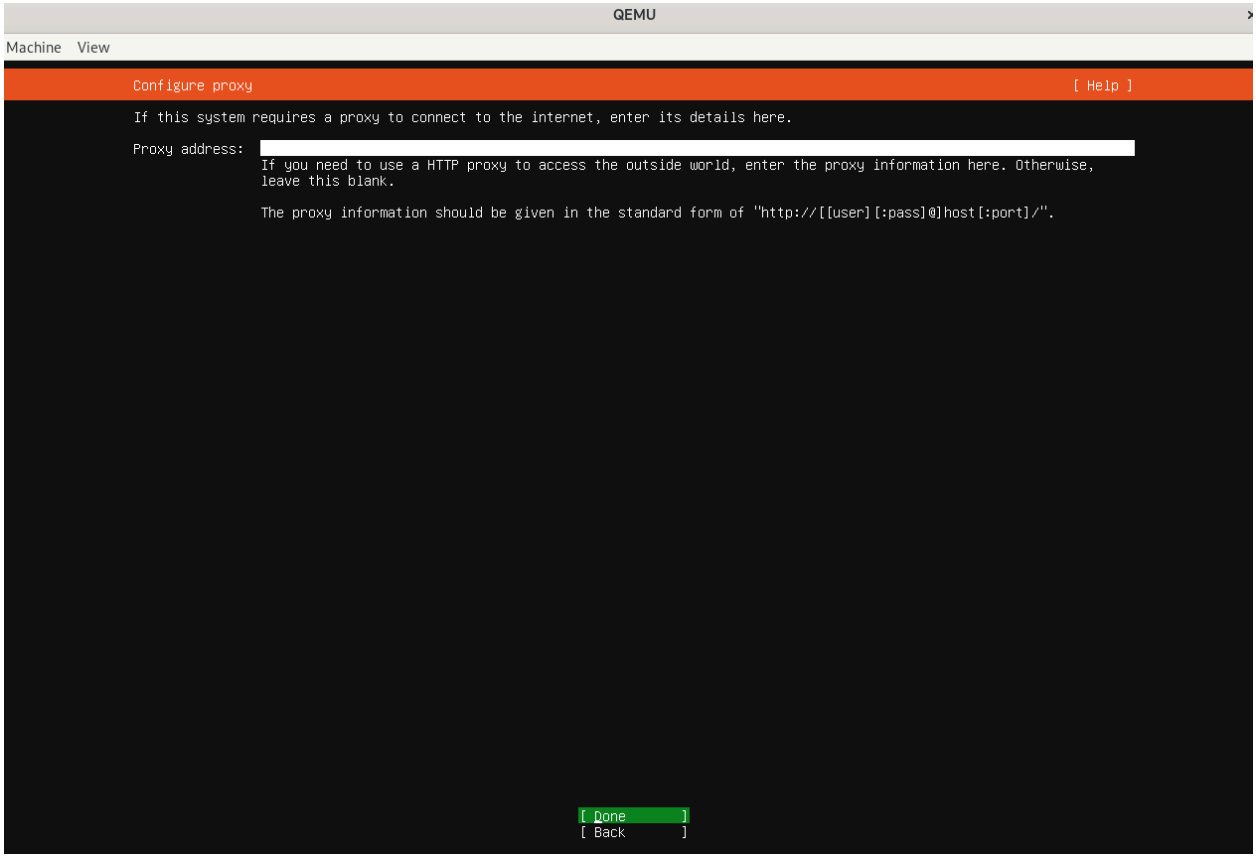
```
[  111.028223] cloud-init[1208]: Your public key has been saved in /etc/ssh/ssh_host_dsa_key.pub
[  111.046449] cloud-init[1208]: The key fingerprint is:
[  111.054050] cloud-init[1208]: SHA256:cKdX/BKrv3QgCBN31CGo38EfZIr32kWnKHQ2+ZYxwVI root@ubuntu-server
[  111.068151] cloud-init[1208]: The key's randomart image is:
[  111.084106] cloud-init[1208]: +---[DSA 1024]----+
[  111.099245] cloud-init[1208]: |      . .o+..+E  |
[  111.106586] cloud-init[1208]: |       o.. o= o  |
[  111.118698] cloud-init[1208]: |      +..o.++o . |
[  111.136475] cloud-init[1208]: |      .=.+*.*++ .|
[  111.150149] cloud-init[1208]: |       .S+o**.* |
[  111.163296] cloud-init[1208]: |        ..oo+o*  |
[  111.171158] cloud-init[1208]: |          .+.o.  |
[  111.192093] cloud-init[1208]: |          .o..   |
[  111.205212] cloud-init[1208]: |            o.   |
[  111.213154] cloud-init[1208]: +----[SHA256]-----+
[  111.226178] cloud-init[1208]: Generating public/private ecdsa key pair.
[  111.238171] cloud-init[1208]: Your identification has been saved in /etc/ssh/ssh_host_ecdsa_key
[  111.247260] cloud-init[1208]: Your public key has been saved in /etc/ssh/ssh_host_ecdsa_key.pub
[  111.259216] cloud-init[1208]: The key fingerprint is:
[  111.281878] cloud-init[1208]: SHA256:CQQADCsEAQIERAtDXYL+pQdNFJI/UyQvy10TPg/pokU root@ubuntu-server
[  111.290007] cloud-init[1208]: The key's randomart image is:
[  111.297914] cloud-init[1208]: +---[ECDSA 256]---+
[  111.311349] cloud-init[1208]: |^O++=*+. .       |
[  111.323093] cloud-init[1208]: |+=.oooo.. o      |
[  111.340346] cloud-init[1208]: |+.  +.oE B       |
[  111.353214] cloud-init[1208]: |.. ..*=.o.=      |
[  111.362772] cloud-init[1208]: |  . +oo+S. .     |
[  111.374355] cloud-init[1208]: |   o .o .        |
[  111.387892] cloud-init[1208]: |    ..           |
[  111.395626] cloud-init[1208]: |                 |
[  111.412546] cloud-init[1208]: |                 |
[  111.419345] cloud-init[1208]: +----[SHA256]-----+
[  111.435116] cloud-init[1208]: Generating public/private ed25519 key pair.
[  111.443620] cloud-init[1208]: Your identification has been saved in /etc/ssh/ssh_host_ed25519_key
[  111.458767] cloud-init[1208]: Your public key has been saved in /etc/ssh/ssh_host_ed25519_key.pub
[  111.475314] cloud-init[1208]: The key fingerprint is:
[  111.486385] cloud-init[1208]: SHA256:4PkoONrW+vWQqmbAco3ifbQdBW39Csfpr00Tnuwc4W4 root@ubuntu-server
[  111.495470] cloud-init[1208]: The key's randomart image is:
[  111.517657] cloud-init[1208]: +--[ED25519 256]--+
[  111.522917] cloud-init[1208]: |         . .     |
[  111.530647] cloud-init[1208]: |        . o .    |
[  111.543925] cloud-init[1208]: |       . o . o   |
[  111.558605] cloud-init[1208]: |    . . o o + .  |
[  111.573769] cloud-init[1208]: | ..o. o S + .o   |
[  111.597706] cloud-init[1208]: |o =+.o + . o+ +  |
[  111.611393] cloud-init[1208]: |.+o.= = *   .O   |
[  111.639329] cloud-init[1208]: | . oo= + o  *Eo  |
[  111.643618] cloud-init[1208]: |   o+oo   ..o=   |
[  111.674484] cloud-init[1208]: +----[SHA256]-----+
```

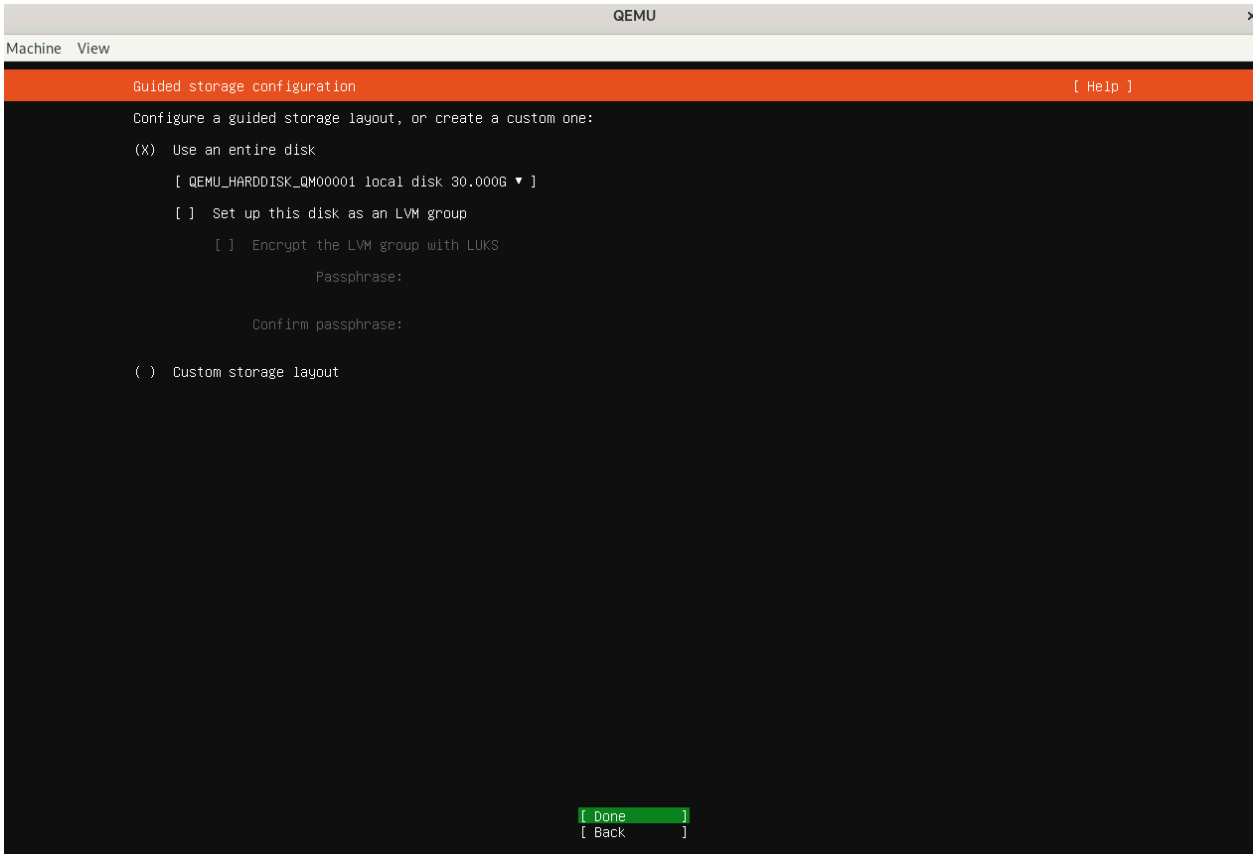


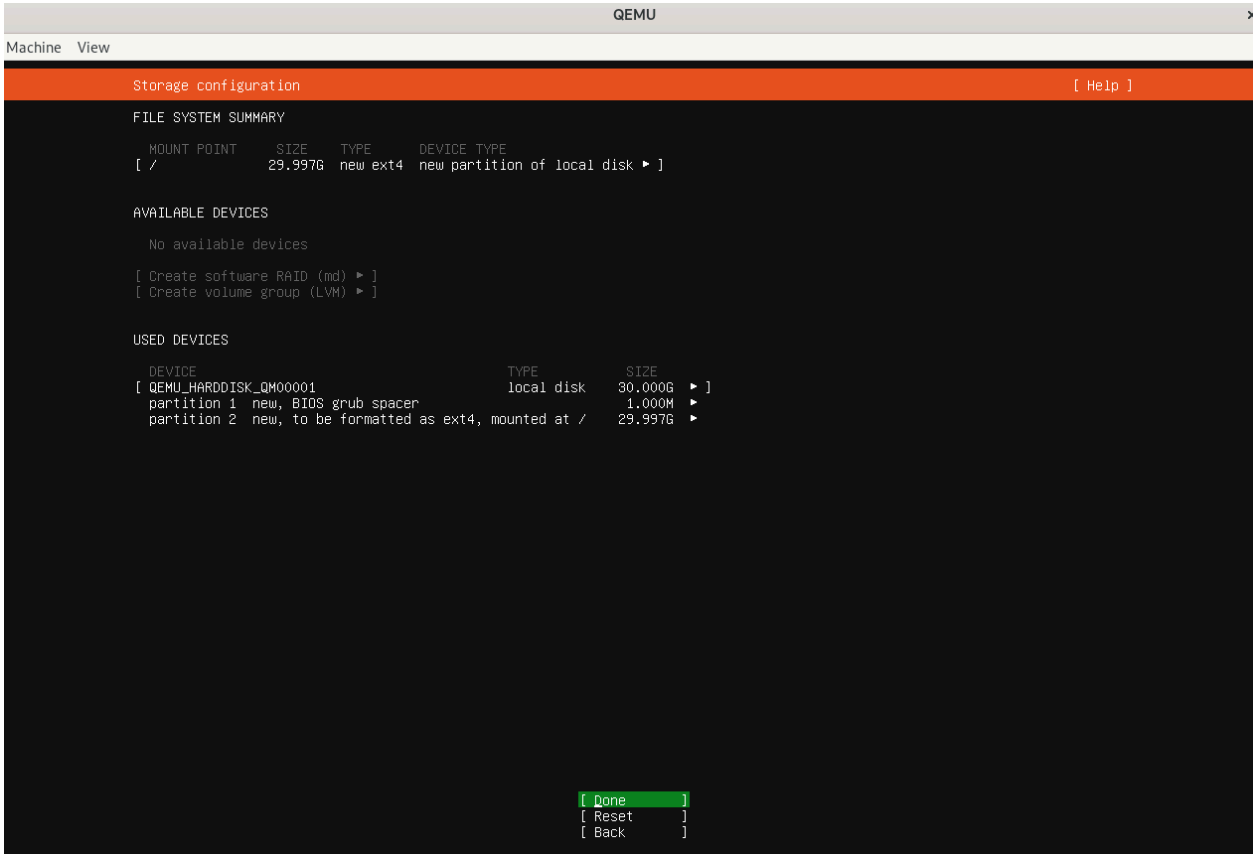

QEMU
Machine View
Installer update available
[Help]
Version 23.10.1 of the installer is now available (22.07.2 is currently running).
You can read the release notes for each version at:
https://github.com/canonical/subiquity/releases
If you choose to update, the update will be downloaded and the installation will continue from here.
[Update to the new installer]
[Continue without updating]
[Back]

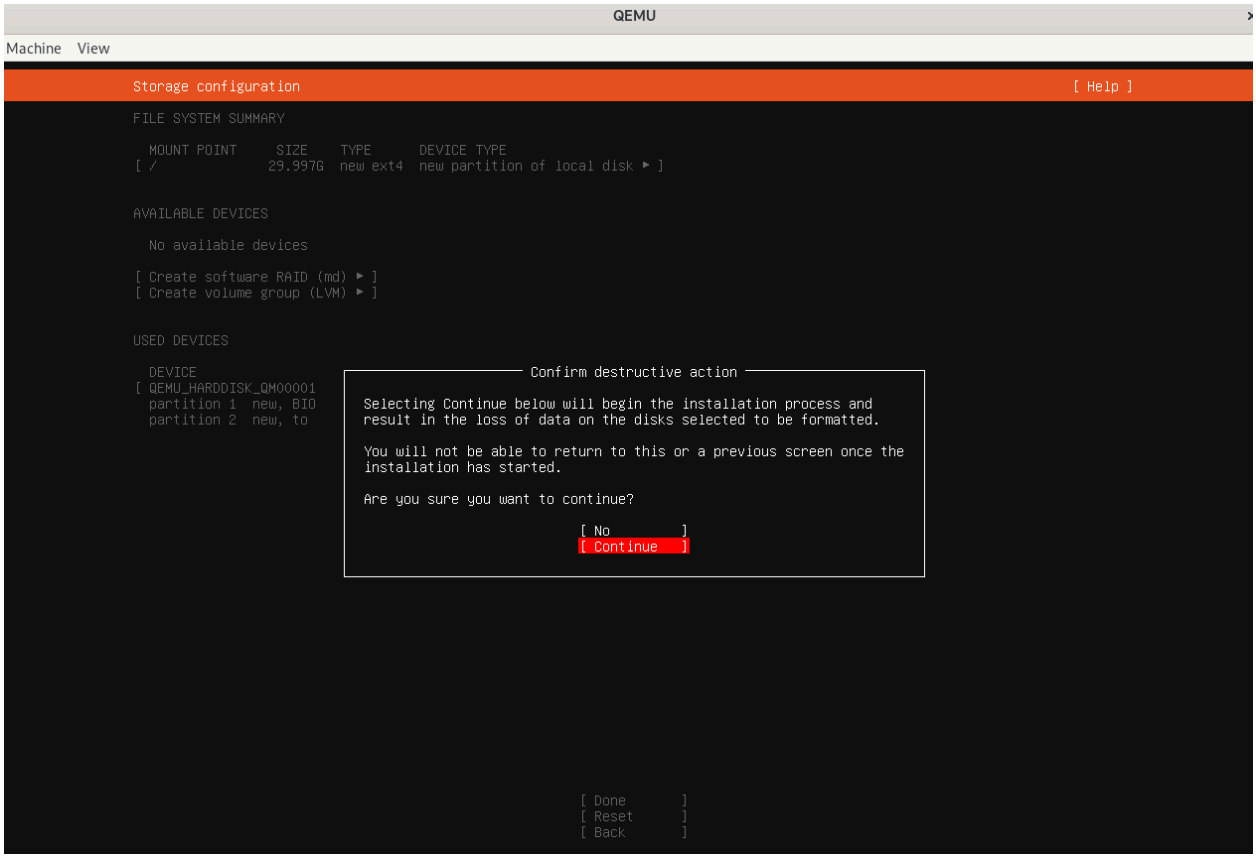

QEMU
Machine View
Keyboard configuration
[Help]
Please select your keyboard layout below, or select "Identify keyboard" to detect your layout automatically.
Layout: [English (US) ▾]
Variant: [English (US) ▾]
[Identify keyboard]
[Done]
[Back]

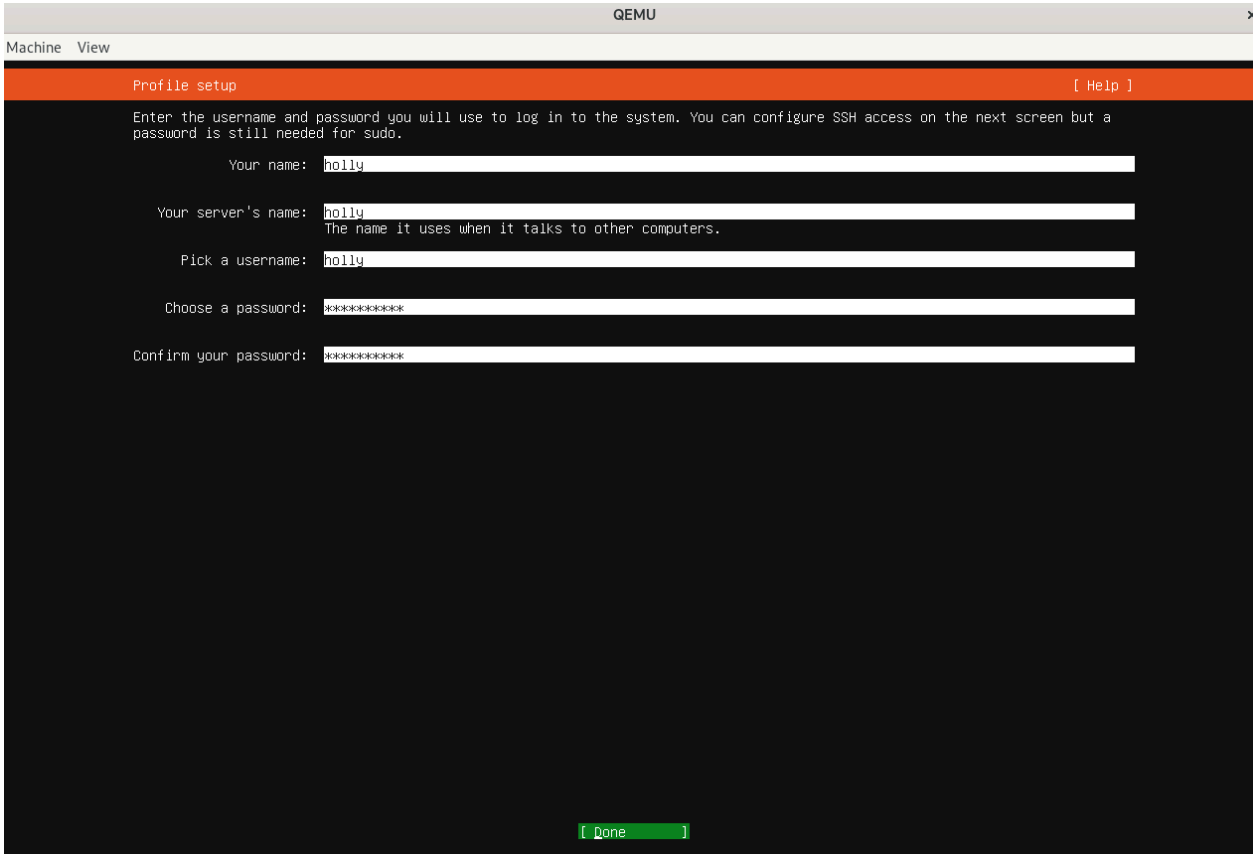

QEMU
Machine View
Network connections
[Help]
Configure at least one interface this server can use to talk to other machines, and which preferably provides sufficient access for updates.
NAME TYPE NOTES
[ens3 eth - ▸]
disabled
52:54:00:12:34:56 / Intel Corporation / 82540EM Gigabit Ethernet Controller (QEMU Virtual Machine)
[Create bond ▸]
[Continue without network]
[Back]

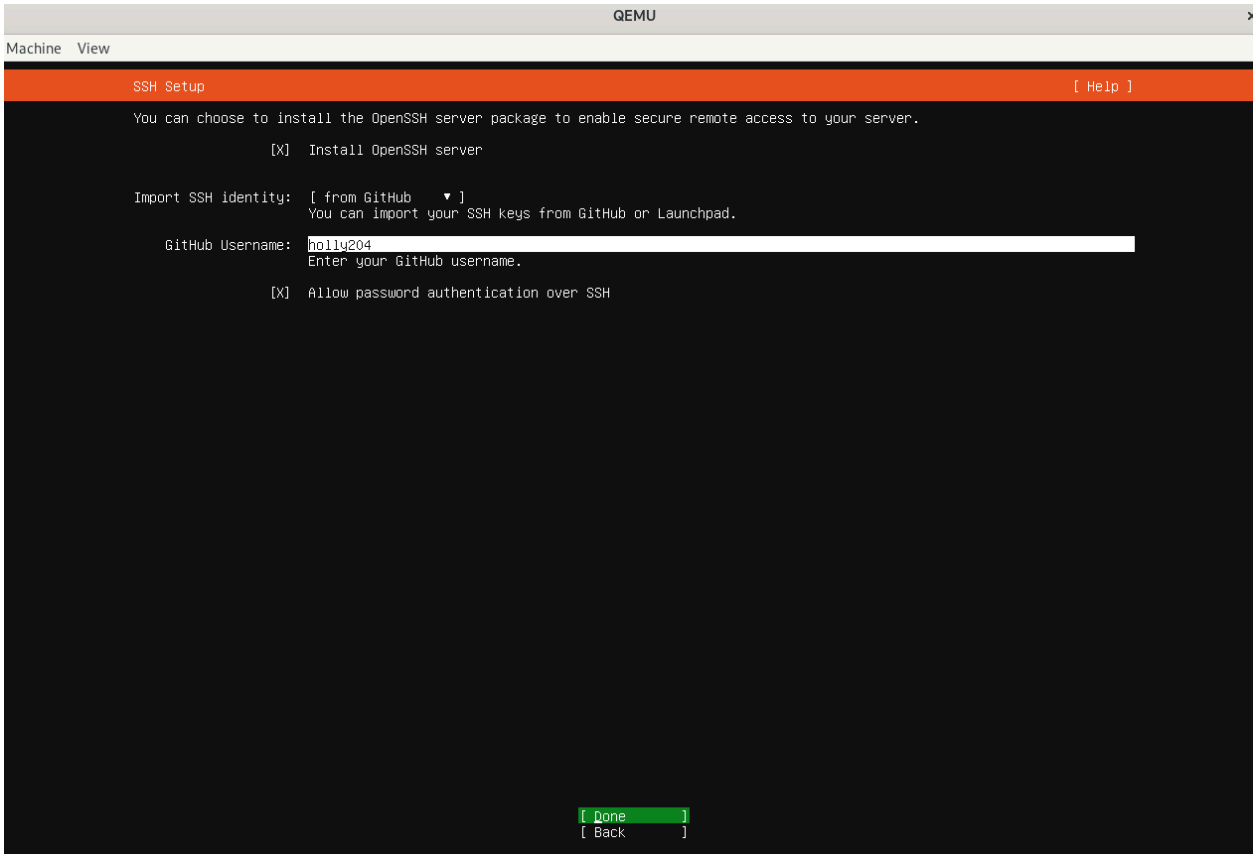

QEMU
Machine View
Configure proxy
[Help]
If this system requires a proxy to connect to the internet, enter its details here.
Proxy address:
If you need to use a HTTP proxy to access the outside world, enter the proxy information here. Otherwise, leave this blank.
The proxy information should be given in the standard form of "http://[[user][:pass]@]host[:port]/".
[Done]
[Back]

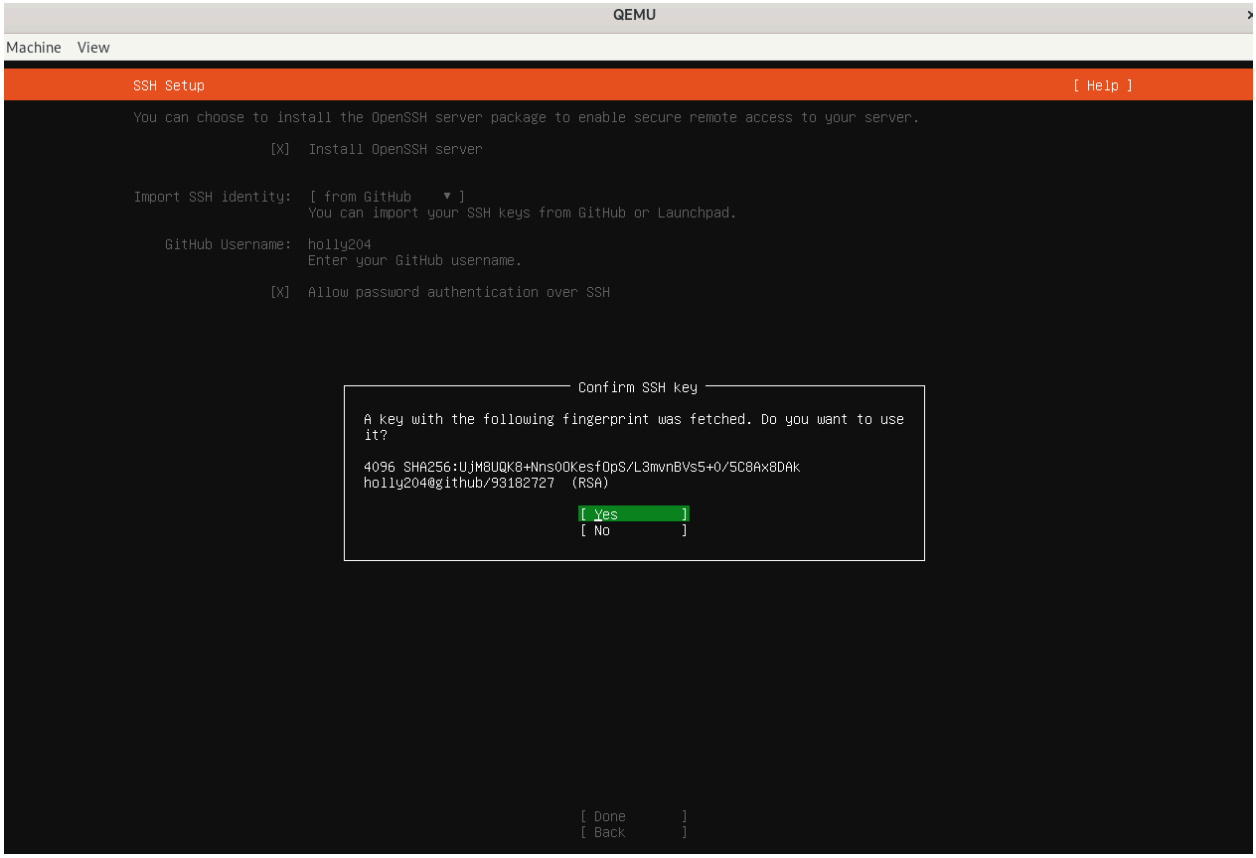

QEMU
Machine View
Guided storage configuration [Help]
Configure a guided storage layout, or create a custom one:
(X) Use an entire disk
[QEMU_HARDDISK_QM00001 local disk 30.000G ▾]
[ ] Set up this disk as an LVM group
[ ] Encrypt the LVM group with LUKS
Passphrase:
Confirm passphrase:
( ) Custom storage layout
[Done]
[Back]

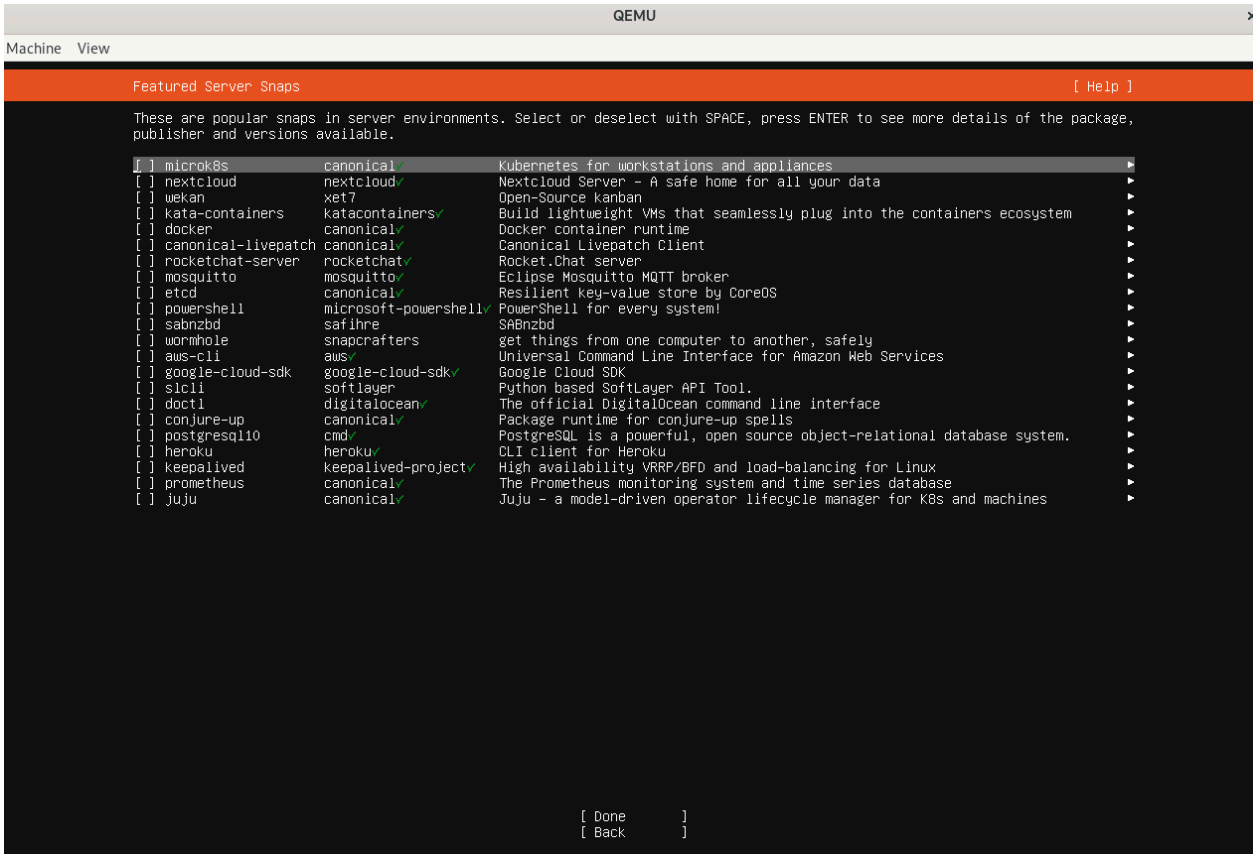

QEMU
Machine View
Storage configuration [Help]
FILE SYSTEM SUMMARY
MOUNT POINT SIZE TYPE DEVICE TYPE
[/ 29.997G new ext4 new partition of local disk ▸]
AVAILABLE DEVICES
No available devices
[Create software RAID (md) ▸]
[Create volume group (LVM) ▸]
USED DEVICES
DEVICE TYPE SIZE
[QEMU_HARDDISK_QM00001 local disk 30.000G ▸]
partition 1 new, BIOS grub spacer 1.000M ▸
partition 2 new, to be formatted as ext4, mounted at / 29.997G ▸
[Done]
[Reset]
[Back]

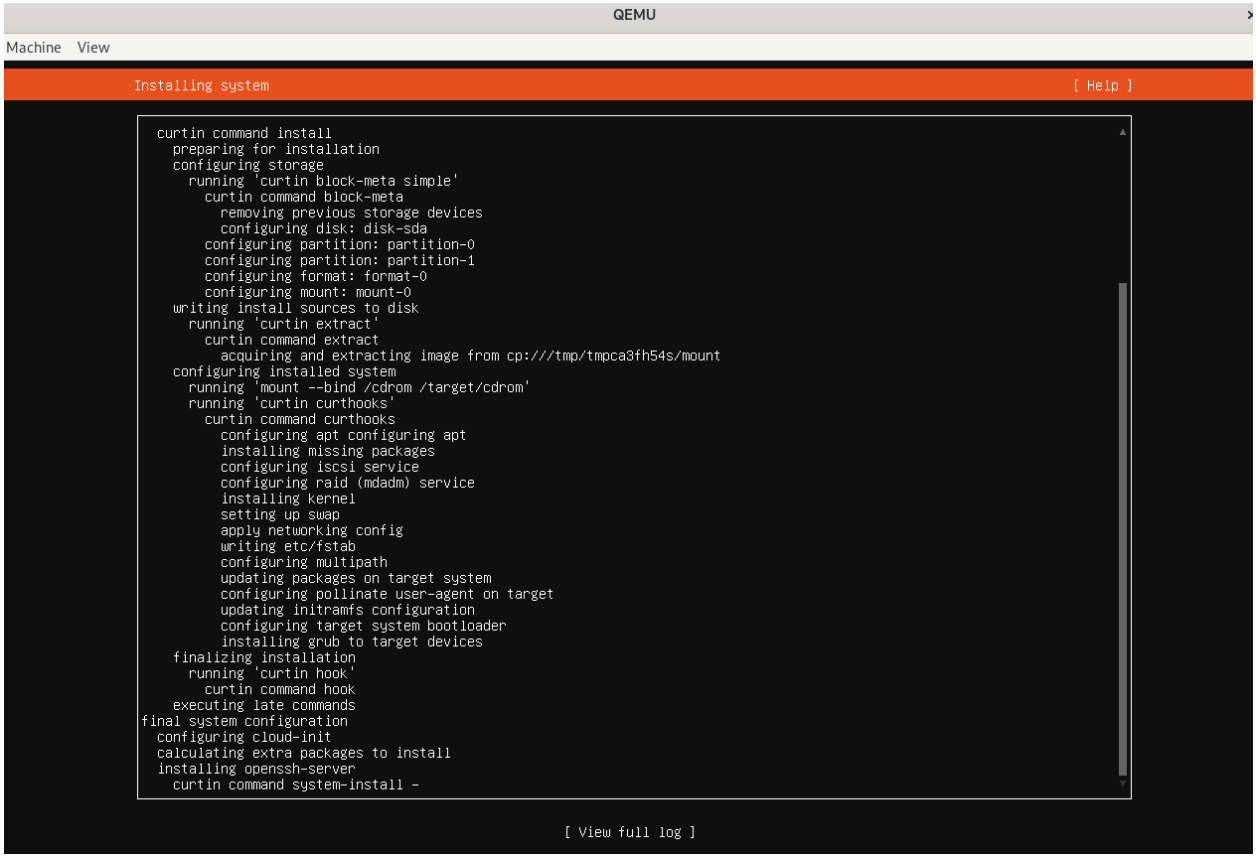

QEMU
Machine View
Storage configuration [Help]
FILE SYSTEM SUMMARY
MOUNT POINT SIZE TYPE DEVICE TYPE
[/ 29.997G new ext4 new partition of local disk ▸]
AVAILABLE DEVICES
No available devices
[Create software RAID (md) ▸]
[Create volume group (LVM) ▸]
USED DEVICES
DEVICE
[QEMU_HARDDISK_QM00001
partition 1 new, BIO
partition 2 new, to
Confirm destructive action
Selecting Continue below will begin the installation process and result in the loss of data on the disks selected to be formatted.
You will not be able to return to this or a previous screen once the installation has started.
Are you sure you want to continue?
[No]
[Continue]
[Done]
[Reset]
[Back]

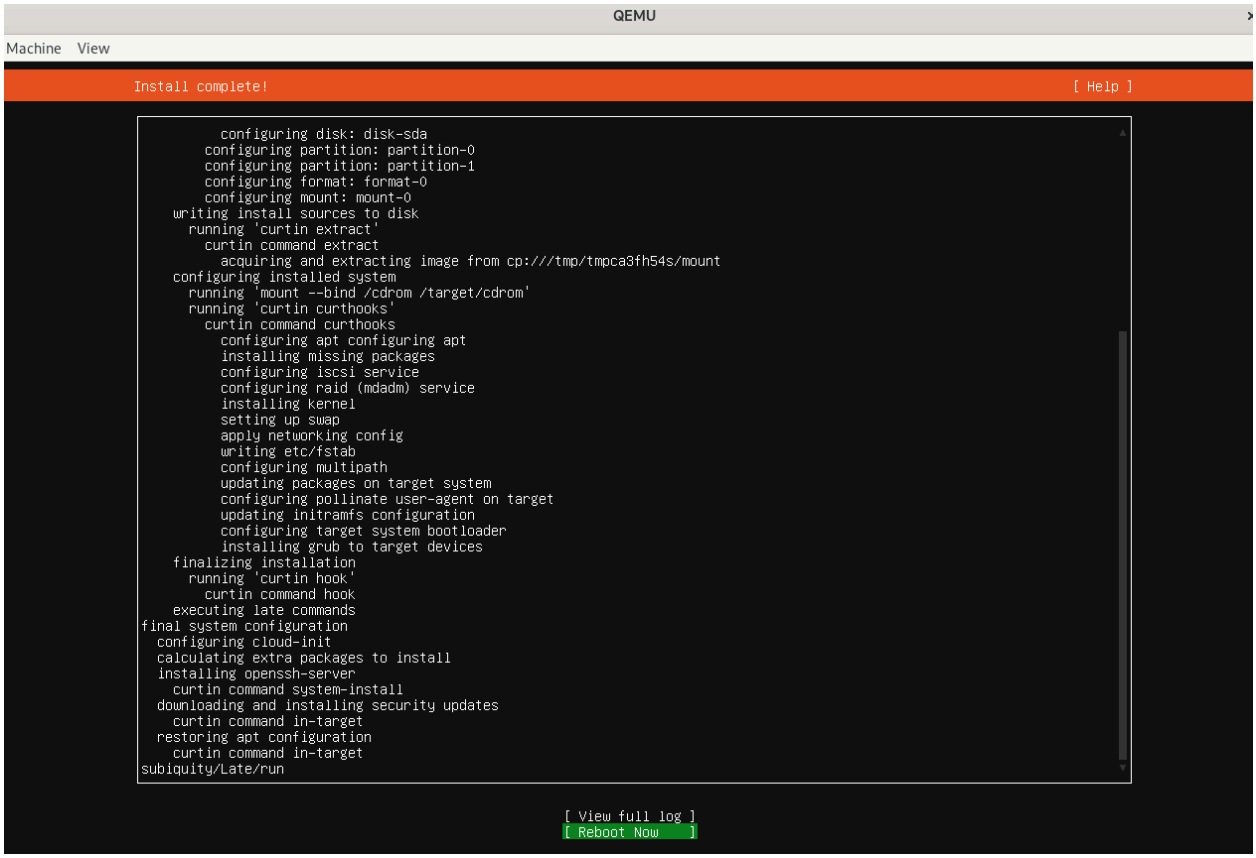

QEMU
Machine View
Profile setup [Help]
Enter the username and password you will use to log in to the system. You can configure SSH access on the next screen but a password is still needed for sudo.
Your name: holly
Your server's name: holly
The name it uses when it talks to other computers.
Pick a username: holly
Choose a password: **********
Confirm your password: **********
[Done]

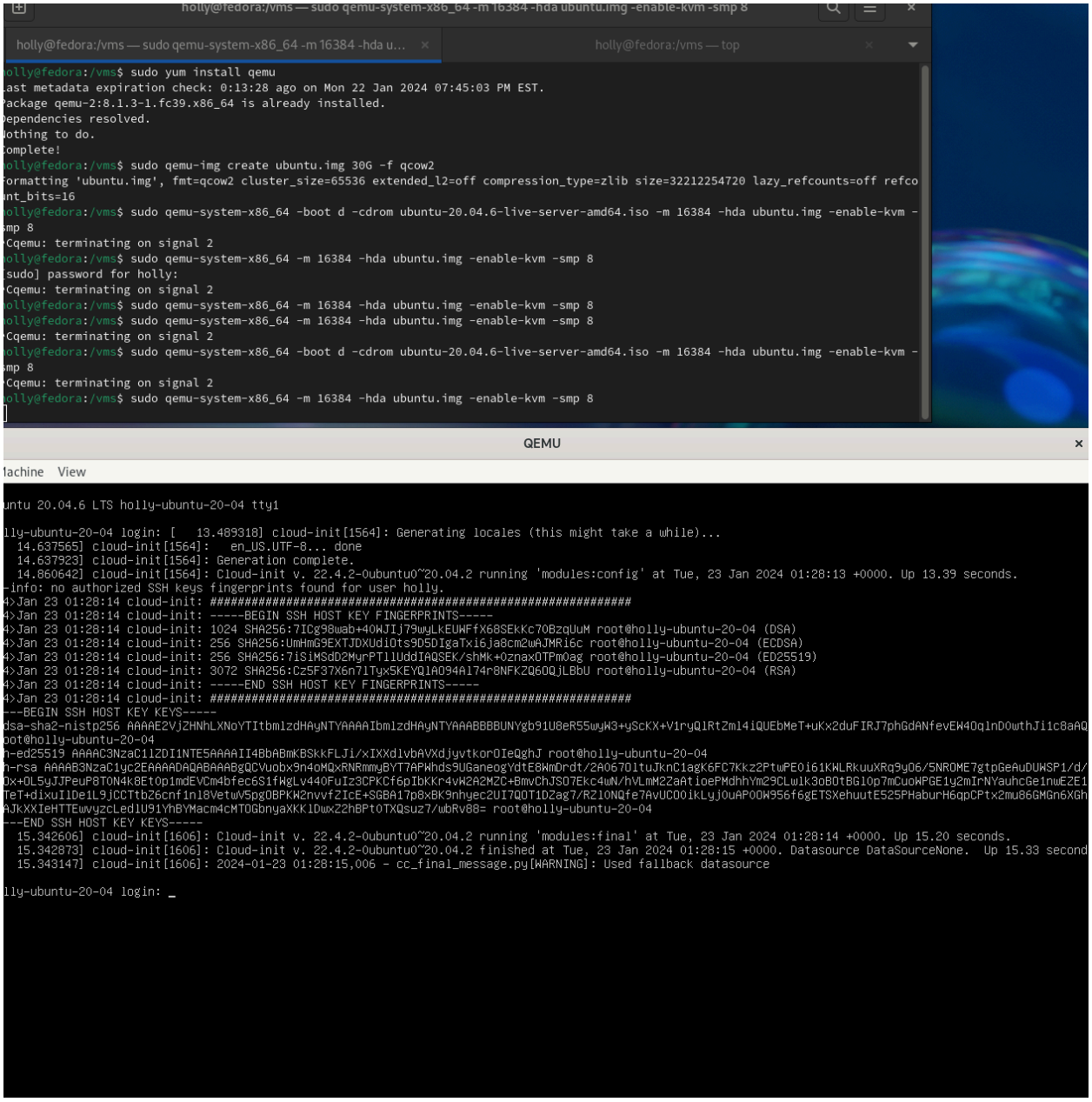

QEMU
Machine View
SSH Setup
[Help]
You can choose to install the OpenSSH server package to enable secure remote access to your server.
[X] Install OpenSSH server
Import SSH identity: [from GitHub ▾]
You can import your SSH keys from GitHub or Launchpad.
GitHub Username: holly204
Enter your GitHub username.
[X] Allow password authentication over SSH
[Done]
[Back]


QEMU
Machine View
SSH Setup
[Help]
You can choose to install the OpenSSH server package to enable secure remote access to your server.
[X] Install OpenSSH server
Import SSH identity: [from GitHub ▾]
You can import your SSH keys from GitHub or Launchpad.
GitHub Username: holly204
Enter your GitHub username.
[X] Allow password authentication over SSH
Confirm SSH key
A key with the following fingerprint was fetched. Do you want to use it?
4096 SHA256:UjM8UQK8+NnsOOKesfOpS/L3mvnBVs5+0/5C8Ax8DAk
holly204@github/93182727 (RSA)
[Yes]
[No]
[Done]
[Back]


QEMU
Machine View
Featured Server Snaps
[Help]
These are popular snaps in server environments. Select or deselect with SPACE, press ENTER to see more details of the package, publisher and versions available.
[ ] microk8s canonical✓ Kubernetes for workstations and appliances
[ ] nextcloud nextcloud✓ Nextcloud Server - A safe home for all your data
[ ] wekan xet7 Open-Source kanban
[ ] kata-containers katacontainers✓ Build lightweight VMs that seamlessly plug into the containers ecosystem
[ ] docker canonical✓ Docker container runtime
[ ] canonical-livepatch canonical✓ Canonical Livepatch Client
[ ] rocketchat-server rocketchat✓ Rocket.Chat server
[ ] mosquitto mosquitto✓ Eclipse Mosquitto MQTT broker
[ ] etcd canonical✓ Resilient key-value store by CoreOS
[ ] powershell microsoft-powershell✓ PowerShell for every system!
[ ] sabnzbd safihre SABnzbd
[ ] wormhole snapcrafters get things from one computer to another, safely
[ ] aws-cli aws✓ Universal Command Line Interface for Amazon Web Services
[ ] google-cloud-sdk google-cloud-sdk✓ Google Cloud SDK
[ ] slcli softlayer Python based SoftLayer API Tool.
[ ] doctl digitalocean✓ The official DigitalOcean command line interface
[ ] conjure-up canonical✓ Package runtime for conjure-up spells
[ ] postgresql10 cmd✓ PostgreSQL is a powerful, open source object-relational database system.
[ ] heroku heroku✓ CLI client for Heroku
[ ] keepalived keepalived-project✓ High availability VRRP/BFD and load-balancing for Linux
[ ] prometheus canonical✓ The Prometheus monitoring system and time series database
[ ] juju canonical✓ Juju - a model-driven operator lifecycle manager for K8s and machines
[Done]
[Back]


QEMU
Machine View
Installing system
[Help]
curtin command install
preparing for installation
configuring storage
running 'curtin block-meta simple'
curtin command block-meta
removing previous storage devices
configuring disk: disk-sda
configuring partition: partition-0
configuring partition: partition-1
configuring format: format-0
configuring mount: mount-0
writing install sources to disk
running 'curtin extract'
curtin command extract
acquiring and extracting image from cp:///tmp/tmpca3fh54s/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 -
[View full log]


QEMU
Machine View
Install complete!
[Help]
configuring disk: disk-sda
configuring partition: partition-0
configuring partition: partition-1
configuring format: format-0
configuring mount: mount-0
writing install sources to disk
running 'curtin extract'
curtin command extract
acquiring and extracting image from cp:///tmp/tmpca3fh54s/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
downloading and installing security updates
curtin command in-target
restoring apt configuration
curtin command in-target
subiquity/Late/run
[View full log]
[Reboot Now]



Reboot

After finishing the installation, just reboot it and log in.

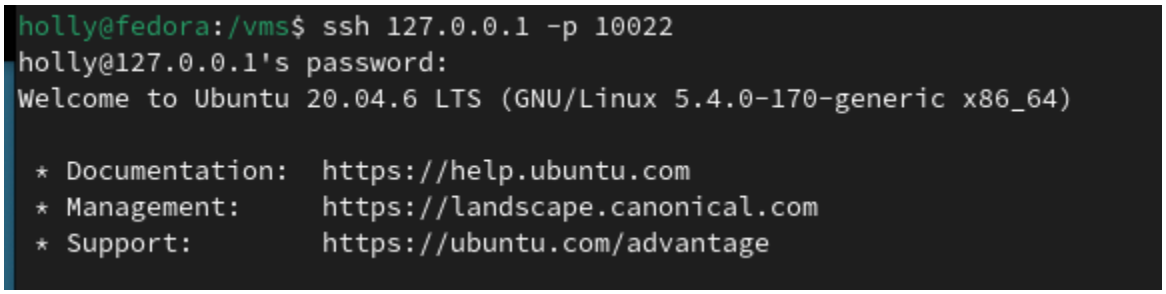
Connect QEMU via SSH

I would like to connect the QEMU via SSH, so I use the following command:

```
$  sudo qemu-system-x86_64 -m 16384 -hda ubuntu.img -enable-kvm -smp 8 -net nic -net user,hostfwd=tcp::10022-:22
```

Forwarding the SSH port 22 to 10022
Now I can connect the QEMU via ssh using the following command:

```
$ ssh 127.0.0.1 -p 10022
```



OS Virtualization (Docker) Setup

Installing Docker on Linux

Follow the instruction on the following website :[Install Docker Engine on Fedora]

Set up the repository

Run the following command:

```
$ sudo dnf -y install dnf-plugins-core
$ sudo dnf config-manager --add-repo
https://download.docker.com/linux/fedora/docker-ce.repo
```

Install the latest version

Just run the command:

```
$ sudo dnf install docker-ce docker-ce-cli containerd.io docker-buildx-plugin
docker-compose-plugin
```

Start Docker

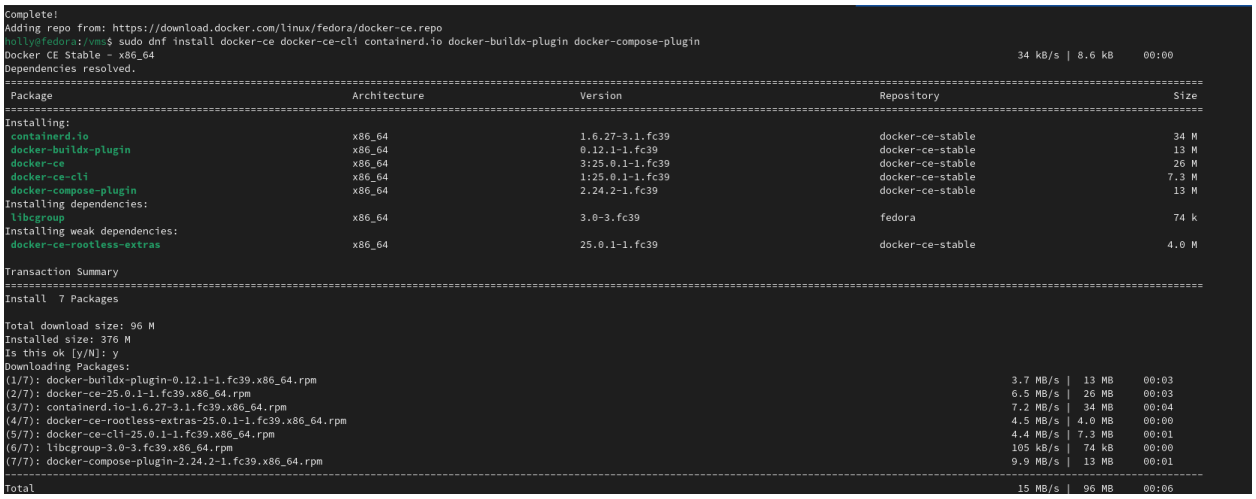
Run the command

```
$  sudo systemctl start docker
```

Verify

Verify that the Docker Engine installation is successful by running the `hello-world` image.

```
$ sudo docker run hello-world
```



Check Docker

To check docker, I also run the command: docker ps, and get the permission denied warning. I run the command

```
$ sudo chmod 777 /var/run/docker.sock
```

```
holly@fedora:/vms$ docker ps
permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Get "http://%2Fvar%2Frun%2Fdocker.sock/v1.24/containers/json": dial unix /var/run/docker.sock: connect: pe
 denied
holly@fedora:/vms$ sudo chmod 777 /var/run/docker.sock
holly@fedora:/vms$ docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
holly@fedora:/vms$ sudo docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
c1ec31eb5944: Pull complete
Digest: sha256:4bd78111b6914a99dbc560e6a20eab57ff6655aea4a80c50b0c5491968cbc2e6
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/

holly@fedora:/vms$
```

Now the docker works successfully.

Installing sysbench

For Ubuntu

Just run the following commands:

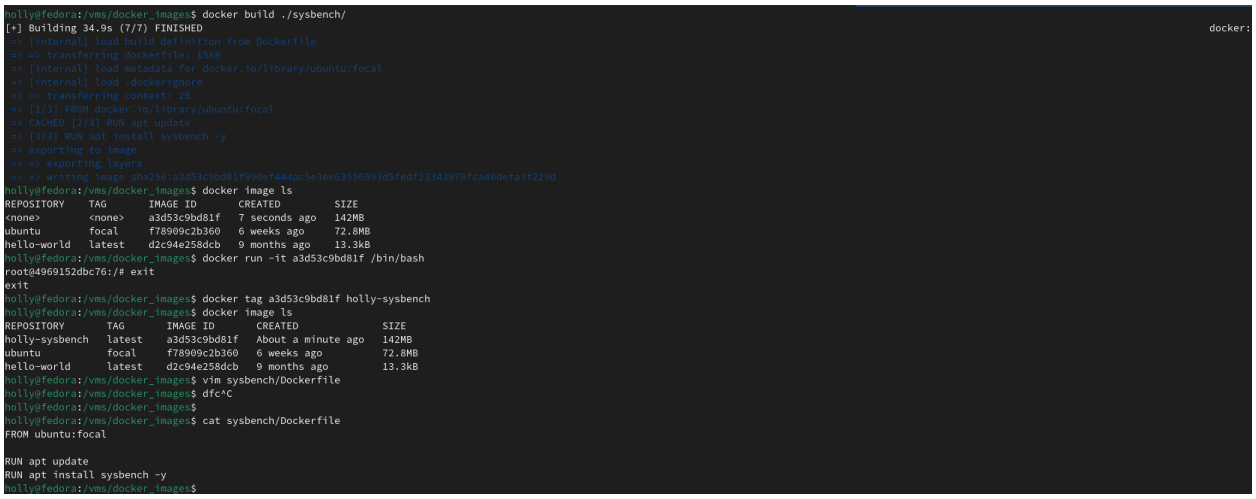
```
$ sudo apt update
$ sudo apt install sysbench
```

```
ils.
holly@holly-ubuntu-20-04:~$ sudo apt install sysbench
Reading package lists... Done
Building dependency tree
Reading state information... Done
sysbench is already the newest version (1.0.18+ds-1).
The following package was automatically installed and is no longer required:
  slirp4netns
Use 'sudo apt autoremove' to remove it.
0 upgraded, 0 newly installed, 0 to remove and 64 not upgraded.
holly@holly-ubuntu-20-04:~$
```

For Docker

Just run the command to build sysbench:

```
$ docker build ./sysbench
```



After building, run

```
$ docker image ls
```

I can see a new image id was just created, but there is no tag on it. I run

```
$ docker tag a3d53c9bd81f holly-sysbench
```

to add the REPOSITORY “holly-sysbench” to it.

To add sysbench to a base image by a dockerfile, by run the following command

```
$ Cat sysbench/Dockerfile
```

And add the following command to the file

```
$ FROM ubuntu:focal

$ RUN apt update
$ RUN apt install sysbench -y
```

To very the sysbench was added to the image, I run command:

```
$ docker run -it a3d53c9bd81f /bin/bash
```

```
RUN apt install sysbench -y
holly@fedora:/vms/docker_images$ docker run -it a3d53c9bd81f /bin/bash
root@330fa61fc063:/# sysbench --num-threads=16 --test=fileio --file-total-size=5G --file-test-mode=rndwr prepare
WARNING: the --test option is deprecated. You can pass a script name or path on the command line without any options.
WARNING: --num-threads is deprecated, use --threads instead
sysbench 1.0.18 (using system LuaJIT 2.1.0-beta3)
```

I can see sysbench information in the just created image.Sysbench was added to the image successfully.

Experiments and Reports

QEMU

Test Case Analysis

To test all the cases, I choose two parameters for each:
1. For CPU Test: I choose cpu-max-prime=2000 and cpu-max-prime=20000
2. For memory test: I choose memory-block-size=1K --memory-total-size=100G --num-threads=4 and memory-block-size=1M --memory-total-size=1T --num-threads=4
3. For fileIO test: I choose file-test-mode=rndrd and file-test-mode=rndwr
4. For different disk image types: I choose raw.img and qcow2.img
5. For CPU: -smp 1 and -smp 4
6. For RAM: -m4096 and -m16384

Then combine them together to get
2 disk drives x 2 QEMU CPU x 2 QEMU Memory x 6 sysbench = 48 cases

	-smp 1 -m 4096 -hdb raw.img	-smp 4 -m 4096 -hdb raw.img	-smp 1 -m 16384 -hdb raw.img	-smp 4 -m 16384 -hdb raw.img	-smp 1 -m 4096 -hdb qcow2.img	-smp 4 -m 4096 -hdb qcow2.img	-smp 1 -m 16384 -hdb qcow2.img
CPU Test-1 --test=cpu --cpu-max-prime=2000 --num-threads=4 run							
CPU Test-2 --test=cpu --cpu-max-prime=20000 --num-threads=4 run							
Memory Test-1 --test=memory --memory-block-size=1K --memory-total-size=100G --num-threads=4							

run							
Memory Test-2 --test=memory --memory-block-size=1M --memory-total-size=1T --num-threads=4 run							
FileIO Test-1 --num-threads=16 --test=fileio --file-total-size=5G --file-test-mode=rndrd							
FileIO Test-2 --num-threads=16 --test=fileio --file-total-size=5G --file-test-mode=rndwr							

Perform Test cases

In order to repeat the sysbench measurement 10 times for each test case, I created a bash script “run-sysbench.sh”to automate the experiment.
Please see the following commands

```
loops=10
current=0
while [[ $current -lt $loops ]];
do
    echo "Round: $current"

    # CPU-Test-1
    sysbench --test=cpu --cpu-max-prime=20000 --num-threads=4 run | tee cpu-test-1-$current.txt

    # CPU-Test-2
    sysbench --test=cpu --cpu-max-prime=20000 --num-threads=4 run | tee cpu-test-2-$current.txt

    # Memory-Test-1
    sysbench --test=memory --memory-block-size=1K --memory-total-size=100G --num-threads=4 run | tee memory-test-1-$current.txt

    # Memory-Test-2
    sysbench --test=memory --memory-block-size=1M --memory-total-size=1T --num-threads=4 run | tee memory-test-2-$current.txt

    # File Read
    sysbench --num-threads=16 --test=fileio --file-total-size=5G --file-test-mode=rndrd prepare &&
echo3 > /proc/sys/vm/drop_caches
    sysbench --num-threads=16 --test=fileio --file-total-size=5G --file-test-mode=rndrd run | tee fileio-test-rndrd-$current.txt &&
echo3 > /proc/sys/vm/drop_caches
    sysbench --num-threads=16 --test=fileio --file-total-size=5G --file-test-mode=rndrd cleanup &&
echo3 > /proc/sys/vm/drop_caches


    # File Write
    sysbench --num-threads=16 --test=fileio --file-total-size=5G --file-test-mode=rndwr prepare &&
echo 3 > /proc/sys/vm/drop_caches
    sysbench --num-threads=16 --test=fileio --file-total-size=5G --file-test-mode=rndwr run | tee fileio-test-rndwr-$current.txt &&
echo 3 > /proc/sys/vm/drop_caches
    sysbench --num-threads=16 --test=fileio --file-total-size=5G --file-test-mode=rndwr cleanup &&
echo 3 > /proc/sys/vm/drop_caches

    current=$((current+1))
done
~
```

For each time, we have to restart the QEMU using different commands,

```
#manurally run the qemu for each time
sudo qemu-system-x86_64 -m 4096 -hda ubuntu.img -hdb raw.img -enable-kvm
```

-smp 1 -net nic -net user,hostfwd=tcp::10022-:22

sudo qemu-system-x86_64 -m 4096 -hda ubuntu.img -hdb raw.img -enable-kvm -smp 4 -net nic -net user,hostfwd=tcp::10022-:22

sudo qemu-system-x86_64 -m 16384 -hda ubuntu.img -hdb raw.img -enable-kvm -smp 1 -net nic -net user,hostfwd=tcp::10022-:22

sudo qemu-system-x86_64 -m 16384 -hda ubuntu.img -hdb raw.img -enable-kvm -smp 4 -net nic -net user,hostfwd=tcp::10022-:22

sudo qemu-system-x86_64 -m 4096 -hda ubuntu.img -hdb qcow2.img -enable-kvm -smp 1 -net nic -net user,hostfwd=tcp::10022-:22

sudo qemu-system-x86_64 -m 4096 -hda ubuntu.img -hdb qcow2.img -enable-kvm -smp 4 -net nic -net user,hostfwd=tcp::10022-:22

sudo qemu-system-x86_64 -m 16384 -hda ubuntu.img -hdb qcow2.img -enable-kvm -smp 1 -net nic -net user,hostfwd=tcp::10022-:22

sudo qemu-system-x86_64 -m 16384 -hda ubuntu.img -hdb qcow2.img -enable-kvm -smp 4 -net nic -net user,hostfwd=tcp::10022-:22

```
^Cqemu: terminating on signal 2
holly@fedora:/vms$ sudo qemu-system-x86_64 -m 4096 -hda ubuntu.img -hdb raw.img -enable-kvm -smp 1 -net nic -net user,hostfwd=tcp::10022-:22
WARNING: Image format was not specified for 'raw.img' and probing guessed raw.
         Automatically detecting the format is dangerous for raw images, write operations on block 0 will be restricted.
         Specify the 'raw' format explicitly to remove the restrictions.
holly@fedora:/vms$ sudo qemu-system-x86_64 -m 4096 -hda ubuntu.img -hdb raw.img -enable-kvm -smp 4 -net nic -net user,hostfwd=tcp::10022-:22
[sudo] password for holly:
WARNING: Image format was not specified for 'raw.img' and probing guessed raw.
         Automatically detecting the format is dangerous for raw images, write operations on block 0 will be restricted.
         Specify the 'raw' format explicitly to remove the restrictions.
holly@fedora:/vms$ sudo qemu-system-x86_64 -m 16384 -hda ubuntu.img -hdb raw.img -enable-kvm -smp 4 -net nic -net user,hostfwd=tcp::10022-:22
[sudo] password for holly:
WARNING: Image format was not specified for 'raw.img' and probing guessed raw.
         Automatically detecting the format is dangerous for raw images, write operations on block 0 will be restricted.
         Specify the 'raw' format explicitly to remove the restrictions.
holly@fedora:/vms$ sudo qemu-system-x86_64 -m 16384 -hda ubuntu.img -hdb raw.img -enable-kvm -smp 1 -net nic -net user,hostfwd=tcp::10022-:22
[sudo] password for holly:
WARNING: Image format was not specified for 'raw.img' and probing guessed raw.
         Automatically detecting the format is dangerous for raw images, write operations on block 0 will be restricted.
         Specify the 'raw' format explicitly to remove the restrictions.
holly@fedora:/vms$ ls
docker_images  lost+found  qcow2.img  raw.img  ubuntu-20.04.6-live-server-amd64.iso  ubuntu.img
```

```
holly@fedora:/vms$ sudo qemu-system-x86_64 -m 4096 -hda ubuntu.img -hdb qcow2.img -enable-kvm -smp 1 -net nic -net user,hostfwd=tcp::10022-:22
[sudo] password for holly:
holly@fedora:/vms$ sudo qemu-system-x86_64 -m 4096 -hda ubuntu.img -hdb qcow2.img -enable-kvm -smp 4 -net nic -net user,hostfwd=tcp::10022-:22
[sudo] password for holly:
holly@fedora:/vms$ sudo qemu-system-x86_64 -m 16384 -hda ubuntu.img -hdb qcow2.img -enable-kvm -smp 1 -net nic -net user,hostfwd=tcp::10022-:22
[sudo] password for holly:
holly@fedora:/vms$ sudo qemu-system-x86_64 -m 16384 -hda ubuntu.img -hdb qcow2.img -enable-kvm -smp 4 -net nic -net user,hostfwd=tcp::10022-:22
[sudo] password for holly:
```

Then, reconnect to the QEMU by ssh,

```
holly@holly-ubuntu-20-04:~/sysbench-sdb/sysbench-g$ Connection to 127.0.0.1 closed by remote host.
Connection to 127.0.0.1 closed.
holly@fedora:/vms$ ssh 127.0.0.1 -p 10022
holly@127.0.0.1's password:
Permission denied, please try again.
holly@127.0.0.1's password:
Welcome to Ubuntu 20.04.6 LTS (GNU/Linux 5.4.0-170-generic x86_64)
```

When changing to different type of disk image from wo raw.img to qcows.img, we need format sdb, by running the following command:

```
sudo mkfs.ext4 /dev/sdb
```

mount /dev/sdb sysbench-sdb by run the command:

```
$  sudo mount /dev/sdb sysbench-sdb
```

```
Last login: Mon Jan 29 19:21:41 2024 from 10.0.2.2
holly@holly-ubuntu-20-04:~$ sudo mount /dev/sdb sysbench-sdb
[sudo] password for holly:
holly@holly-ubuntu-20-04:~$
```

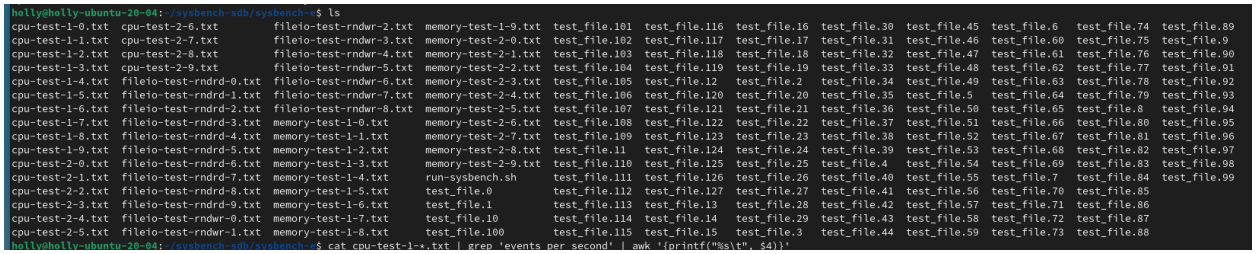
and run the bash by run the command:

```
$ bash run-sysbench.sh
```

```
holly@holly-ubuntu-20-04:~/sysbench-sdb/sysbench-h$ bash run-sysbench.sh
Round: 0
WARNING: the --test option is deprecated. You can pass a script name or path on the command line without any options.
WARNING: --num-threads is deprecated, use --threads instead
sysbench 1.0.18 (using system LuaJIT 2.1.0-beta3)

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


Prime numbers limit: 20000

Initializing worker threads...

Threads started!
```

After that, It will create lots of data files for CPU, Memory and FileIO test

```
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
```



CPU data file sample:

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


Prime numbers limit: 20000

Initializing worker threads...

Threads started!

CPU speed:
    events per second:  2053.24

General statistics:
    total time:                          10.0019s
    total number of events:              20539

Latency (ms):
         min:                                    1.94
         avg:                                    1.95
         max:                                    2.67
         95th percentile:                        1.96
         sum:                                39998.37

Threads fairness:
    events (avg/stddev):           5134.7500/3.42
    execution time (avg/stddev):   9.9996/0.00
```

Memory data file sample:

```
Running memory speed test with the following options:
  block size: 1KiB
  total size: 102400MiB
  operation: write
  scope: global

Initializing worker threads...

Threads started!

Total operations: 101069958 (10105478.83 per second)

98701.13 MiB transferred (9868.63 MiB/sec)


General statistics:
    total time:                          10.0001s
    total number of events:              101069958

Latency (ms):
         min:                                    0.00
         avg:                                    0.00
         max:                                    0.14
         95th percentile:                        0.00
         sum:                                27807.36

Threads fairness:
    events (avg/stddev):           25267489.5000/528013.17
    execution time (avg/stddev):   6.9518/0.04
```

FileIO data file sample:

```
Doing random read test
Initializing worker threads...

Threads started!


File operations:
    reads/s:                      1459742.98
    writes/s:                     0.00
    fsyncs/s:                     0.00

Throughput:
    read, MiB/s:                  22808.48
    written, MiB/s:               0.00

General statistics:
    total time:                          10.0001s
    total number of events:              14599458

Latency (ms):
         min:                                    0.00
         avg:                                    0.01
         max:                                    8.11
         95th percentile:                        0.01
         sum:                               114085.23

Threads fairness:
    events (avg/stddev):           912466.1250/52033.69
    execution time (avg/stddev):   7.1303/0.05
```

Test Result

Using the following command to get data result

```
$ cat cpu-test-1-*.txt | grep 'events per second' | awk '{printf("%s\t", $4)}'
$ cat cpu-test-2-*.txt | grep 'events per second' | awk '{printf("%s\t", $4)}'
$ cat memory-test-1-*.txt | grep 'Total operations' | awk '{printf("%s\t", $4)}'
$ cat memory-test-2-*.txt | grep 'Total operations' | awk '{printf("%s\t", $4)}'
$ cat fileio-test-rndrd-*.txt | grep 'read, MiB/s:' | awk '{printf("%s\t", $3)}'
$ cat fileio-test-rndwr-*.txt | grep 'written, MiB/s' | awk '{printf("%s\t", $3)}'
```

For raw.img

			Round 1	Round 2	Round 3	Round 4	Round 5	Round 6	Round 7	Round 8	Round 9	Round 10	STD	AVG	MAX	MIN
	CPU Test 1	events per second	530.2	523.16	531.37	526.5	531.28	531.37	529.53	526.95	514.19	518.35	5.981720303	526.29	531.37	514.19
	CPU Test 2	events per second	530.69	526.56	524.42	530.64	529.49	532.02	525.7	512.81	522.1	516.9	6.318626873	525.133	532.02	512.81
	Memory Test 1	operations per second	6343629.92	6343592.34	6321634.53	6360655.98	6355263.47	6337546.19	6207563.95	6064385.48	6252103.35	6105508.51	109087.5898	6269188.372	6360655.98	6064385.48
	Memory Test 2	operations per second	20659.95	20668.52	20653.39	20653.62	20649.1	20616.26	19722.63	19936.77	20348.33	19944.53	373.9284967	20385.31	20668.52	19722.63
-smp 1	fileio-test-rndrd-	read, MiB/s	142.82	142.42	144	142.28	146.94	143.98	139.81	137.57	138.29	142.68	2.825209727	142.079	146.94	137.57
-m 4096	fileio-test-rndwr-	written, MiB/s	18.1	19.18	19.16	19.43	18.27	18.98	18.11	18.25	19.38	19.57	0.5928846618	18.843	19.57	18.1
-hdb raw.img																

			Round 1	Round 2	Round 3	Round 4	Round 5	Round 6	Round 7	Round 8	Round 9	Round 10	STD	AVG	MAX	MIN
	CPU Test 1	events per second	2056.76	2065.89	2082.15	2054.14	2060.49	2062.37	2064.15	2063.74	2057.43	2066.44	7.765093117	2063.356	2082.15	2054.14
	CPU Test 2	events per second	2049.58	2057.08	2080.29	2055.23	2055.89	2065.93	2081.84	2069.51	2052.46	2052.78	11.7265496	2062.059	2081.84	2049.58
	Memory Test 1	operations per second	9365129.14	9583379.63	10314983.39	9886362.2	9226660.04	9175830.21	9634234.46	9367153.95	9972234.56	9901994.54	370402.3706	9642796.212	10314983.39	9175830.21
	Memory Test 2	operations per second	77100.28	77027.99	77951.54	77360.04	77356.35	78228.08	77381.26	76224.93	77350.18	78069.52	582.4532936	77405.017	78228.08	76224.93
-smp 4	fileio-test-rndrd-	read, MiB/s	313.88	334.09	332.81	331.66	342.42	344.72	343.11	345.91	351.46	344.18	10.72450693	338.424	351.46	313.88
-m 4096	fileio-test-rndwr-	written, MiB/s	17.6	19.33	18.62	20.06	18.7	19.25	19.49	18.25	18.58	19.9	0.7654744353	18.978	20.06	17.6
-hdb raw.img																

			Round 1	Round 2	Round 3	Round 4	Round 5	Round 6	Round 7	Round 8	Round 9	Round 10	STD	AVG	MAX	MIN
	CPU Test 1	events per second	516.89	527.07	526.88	525.4	526.46	526.01	534.3	530.92	531.74	532.53	4.955849518	527.82	534.3	516.89
	CPU Test 2	events per second	526.47	525.76	526.92	526.32	526.08	526.33	536.57	523.45	532.52	531.67	4.019846459	528.209	536.57	523.45
	Memory Test 1	operations per second	6289503.5	6304364.86	6298954.34	6317737.67	6306454.25	6278729.22	6425995.76	6271363.76	6386509.62	6356948.4	50293.4964	6323656.138	6425995.76	6271363.76
	Memory Test 2	operations per second	20511.87	20479.2	20473.09	20497.99	20459.89	20482.09	20880.3	20201.98	20720.31	20691.5	184.3987663	20539.822	20880.3	20201.98
-smp 1	fileio-test-rndrd-	read, MiB/s	5728.42	5648.88	5573.01	5700.36	5624.76	5713.9	5799.79	5730.73	5717.34	5663.52	64.07443648	5690.071	5799.79	5573.01
-m 16384	fileio-test-rndwr-	written, MiB/s	19.13	18	16.95	19.12	18.08	18.38	17.27	19.01	19.36	17.3	0.8819926933	18.26	19.36	16.95
-hdb raw.img																

			Round 1	Round 2	Round 3	Round 4	Round 5	Round 6	Round 7	Round 8	Round 9	Round 10	STD	AVG	MAX	MIN
	CPU Test 1	events per second	2076.36	2051.04	2054.7	2067.87	2088.38	2052.16	2060.53	2057.52	2059.49	2076.16	12.37036194	2064.421	2088.38	2051.04
	CPU Test 2	events per second	2058.35	2054.44	2054.25	2079.41	2069.37	2074.43	2054.05	2055.27	2053.18	2061.07	9.578923391	2061.382	2079.41	2053.18
	Memory Test 1	operations per second	10058601.03	9199862.38	8898490.42	9383722.42	9585749.64	9751431.34	8872187.25	9314108.95	9211222.13	9121337.73	372148.7081	9339671.329	10058601.03	8872187.25
	Memory Test 2	operations per second	77389.42	77520.06	76397.13	77136.19	77186.98	77005.6	77207.8	77058.97	76991.48	78009.99	414.1305423	77190.362	78009.99	76397.13
-smp 4	fileio-test-rndrd-	read, MiB/s	18104.16	17944.87	18108.63	17975.34	18210.9	18338.82	18344.14	18246.53	18189.73	18379.95	151.089808	18184.307	18379.95	17944.87
-m 16384	fileio-test-rndwr-	written, MiB/s	18.67	17.73	17.12	18.81	16.71	19.82	19.18	16.62	19.4	17.54	1.161474159	18.16	19.82	16.62
-hdb raw.img																

For qcow2.img

			Round 1	Round 2	Round 3	Round 4	Round 5	Round 6	Round 7	Round 8	Round 9	Round 10	STD	AVG	MAX	MIN
	CPU Test 1	events per second	519.32	528.74	518.09	514.88	514.94	526.55	519.54	530.9	532.68	518.3	6.909930254	522.8488889	532.68	514.88
	CPU Test 2	events per second	517.46	519.39	520.2	518.51	518.55	525.2	517.24	530.89	532	516.97	5.771109079	522.16	532	517.24
	Memory Test 1	operations per second	505121.48	506393.97	497366.2	497866.63	512642.16	485965.17	497000.66	509605.97	522099.72	488648.93	10583.92976	503784.6622	522099.72	485965.17
	Memory Test 2	operations per second	18014.45	17924.74	17656.61	17710.85	18050.06	17893.6	17899.91	18288.95	18490.47	17765.13	263.3308568	17992.18222	18490.47	17656.61
-smp 1	fileio-test-rndrd-	read, MiB/s	292.87	315.59	324.82	324.32	324.56	331.22	335.2	333.62	327.74	338.98	12.84967607	323.3266667	335.2	292.87
-m 4096	fileio-test-rndwr-	written, MiB/s	18.24	18.98	19.48	18.99	18.73	19.05	17.73	18.41	18.85	19.81	0.5184780077	18.71777778	19.48	17.73
-hdb qcow2.img																

			Round 1	Round 2	Round 3	Round 4	Round 5	Round 6	Round 7	Round 8	Round 9	Round 10	STD	AVG	MAX	MIN
	CPU Test 1	events per second	2044.27	2044.53	2039.14	2054.35	2049.45	2045.05	2049.75	2043.82	2044.19	2047.95	4.216494068	2046.25	2054.35	2039.14
	CPU Test 2	events per second	2044.49	2040.15	2042.74	2039.94	2036.01	2039.32	2042.43	2049.49	2058.61	2051.73	6.849780937	2044.491	2058.61	2036.01
	Memory Test 1	operations per second	9514989.9	9446338.81	8999276.18	9227545.54	9625598.64	9880128.69	9505870.11	9048885.12	9673565.41	9662579.78	285031.8013	9458477.818	9880128.69	8999276.18
	Memory Test 2	operations per second	75984.29	75842.56	74143.79	75455.75	75382.95	77273.89	75008.62	76257.19	74995.18	76309.75	869.5657817	75665.397	77273.89	74143.79
-smp 4	fileio-test-rndrd-	read, MiB/s	307.48	321.93	326.57	334.59	335.75	343.12	337.02	336.33	333.24	342.4	10.69758857	331.843	343.12	307.48
-m 4096	fileio-test-rndwr-	written, MiB/s	19.39	19.3	19.78	18.98	19.82	20.04	19.74	19.68	19.6	19.54	0.301995217	19.587	20.04	18.98
-hdb qcow2.img																

			Round 1	Round 2	Round 3	Round 4	Round 5	Round 6	Round 7	Round 8	Round 9	Round 10	STD	AVG	MAX	MIN
	CPU Test 1	events per second	520.18	515.08	518.15	520.49	525.13	516.42	516.37	534.19	515.66	517.42	5.840251992	519.909	534.19	515.08
	CPU Test 2	events per second	517.65	514.78	517.11	520.92	524.72	518.65	515.96	532.76	520.58	522.56	5.258452349	520.569	532.76	514.78
	Memory Test 1	operations per second	6174457.46	6206903.24	6146992.74	6248883.14	6302451.29	6210114.99	6185327.6	6365419.17	6165903.79	6196110.73	67619.21366	6220256.415	6365419.17	6146992.74
	Memory Test 2	operations per second	20039.73	20109.31	19992.38	20283.16	20594.16	19896.7	19907.34	20713.09	20289.78	20222.73	276.5943795	20204.838	20713.09	19896.7
-smp 1	fileio-test-rndrd-	read, MiB/s	5628.82	5512.36	5563.92	5466.11	5498.86	5382.9	5654.77	5466.75	5636.95	5663.17	96.38405337	5547.461	5663.17	5382.9
-m 16384	fileio-test-rndwr-	written, MiB/s	18.91	19.15	19.17	19.55	18.71	19.27	17.42	19.09	18.84	19.16	0.5794067848	18.927	19.55	17.42
-hdb qcow2.img																

			Round 1	Round 2	Round 3	Round 4	Round 5	Round 6	Round 7	Round 8	Round 9	Round 10	STD	AVG	MAX	MIN
	CPU Test 1	events per second	2048.18	2071.21	2061.26	2039.14	2040.97	2040.72	2042.26	2039.36	2030.55	2044.07	11.89485304	2045.772	2071.21	2030.55
	CPU Test 2	events per second	2040.64	2080.62	2054.15	2055.56	2043.35	2036.37	2052.22	2038.88	2038.68	2033.59	13.99227739	2047.406	2080.62	2033.59
	Memory Test 1	operations per second	9563070.22	9736275.43	9369811.47	10074770.32	9134204.54	9259168.23	9635766.88	10146159.15	8978746.73	9536535.94	378452.1848	9543450.891	10146159.15	8978746.73
	Memory Test 2	operations per second	75413.83	77146.93	77601.68	74234.45	75178.38	74805.12	75876.08	4546.68	75673.74	74111.18	22484.5993	68458.807	77601.68	4546.68
-smp 4	fileio-test-rndrd-	read, MiB/s	16909.4	18091.84	18204.87	17112.47	17076.57	16819.11	16572.85	16468.64	16743.7	16411.85	629.1760252	17041.13	18204.87	16411.85
-m 16384	fileio-test-rndwr-	written, MiB/s	15.14	18.98	19.06	18.44	20.03	19.44	19.77	19.3	19.61	19.82	1.421356551	18.959	20.03	15.14
-hdb qcow2.img																

Result Analysis

	-smp 1 -m 4096 -hdb raw.img	-smp 4 -m 4096 -hdb raw.img	-smp 1 -m 16384 -hdb raw.img	-smp 4 -m 16384 -hdb raw.img	-smp 1 -m 4096 -hdb qcow2.img	-smp 4 -m 4096 -hdb qcow2.img	-smp 1 -m 16384 -hdb qcow2.img	-smp 4 -m 16384 -hdb qcow2.img
CPU Test-1 --test=cpu --cpu-max-prime=2000 --num-threads=4 run	526.29	2063.356	527.82	2064.421	522.8488889	2046.25	519.909	2045.772

CPU Test-2 --test=cpu --cpu-max-prime=20000 --num-threads=4 run	525.133	2062.059	528.209	2061.382	522.16	2044.491	520.569	2047.406
Memory Test-1 --test=memory --memory-block-size=1K --memory-total-size=100G --num-threads=4 run	6269188.372	9642796.212	6323656.138	9339671.329	503784.6622	9458477.818	6220256.415	9543450.891
Memory Test-2 --test=memory --memory-block-size=1M --memory-total-size=1T --num-threads=4 run	20385.31	77405.017	20539.822	77190.362	17992.18222	75665.397	20204.838	68458.807
FileIO Test-1 --num-threads=16 --test=fileio --file-total-size=5G --file-test-mode=rndrd	142.079	338.424	5690.071	18184.307	323.3266667	331.843	5547.461	17041.13
FileIO Test-2 --num-threads=16 --test=fileio --file-total-size=5G --file-test-mode=rndwr	18.843	18.978	18.26	18.16	18.71777778	19.587	18.927	18.959

The above data was the average of 10 times for each case.
From the table:

1. cpu-max-prime=2000 is similar to cpu-max-prime=20000, both of them have similar events per second.
2. memory-block-size=1K --memory-total-size=100G has 100 times operations per second than memory-block-size=1M --memory-total-size=1T when having one CPU, and 300 times operations per second when have 4 CPUs
3. For all test cases, FileIO written, MiB/s is similar.
4. For raw.img and qcow2.img, they are similar to each other.
5. All the data in 4 CPUs is 4 times than 1 CPU
6. RAM 4096 and RAM 16384 are similar to each other.

Therefore, the number of CPUs and memory-block-size affects the performance.

Docker

Test Case Analysis

The case is similar to QEMU, except different type of disk image
To test all the cases, I choose two parameters for each:
1. For CPU Test: I choose cpu-max-prime=2000 and cpu-max-prime=20000
2. For memory test: I choose memory-block-size=1K --memory-total-size=100G --num-threads=4 and memory-block-size=1M --memory-total-size=1T --num-threads=4
3. For fileIO test: I choose file-test-mode=rndrd and file-test-mode=rndwr
4. For CPU: cpus=1 and cpus=4
5. For RAM: memory=4096M and memory=16384M

Then combine them together to get
2 QEMU CPU x 2 QEMU Memory x 6 sysbench = 24 cases

	A	B	C	D	E
1		--cpus=1 --memory=4096M	--cpus=4 --memory=4096M	--cpus=1 --memory=1638	--cpus=4 --memory=16384M
2	CPU Test-1 --test=cpu --cpu-max-prime=2000 --num-threads=4 run				
3	CPU Test-2 --test=cpu --cpu-max-prime=20000 --num-threads=4 run				
4	Memory Test-1 --test=memory --memory-block-size=1K --memory-total-size=100G --num-threads=4 run				
5	Memory Test-2 --test=memory --memory-block-size=1M --memory-total-size=1T --num-threads=4 run				
6	FileIO Test-1 --num-threads=16 --test=fileio --file-total-size=5G --file-test-mode=rndrd				
7	FileIO Test-2 --num-threads=16 --test=fileio --file-total-size=5G --file-test-mode=rndwr				

Perform Test cases

In order to repeat the sysbench measurement 10 times for each test case, I created a bash script “run-sysbench.sh”to automate the experiment.
Please see the following commands

```
loops=10
current=0
while [[ $current -lt $loops ]];
do
    echo "Round: $current"

    # CPU-Test-1
    sysbench --test=cpu --cpu-max-prime=20000 --num-threads=4 run | tee cpu-test-1-$current.txt

    # CPU-Test-2
    sysbench --test=cpu --cpu-max-prime=20000 --num-threads=4 run | tee cpu-test-2-$current.txt

    # Memory-Test-1
    sysbench --test=memory --memory-block-size=1K --memory-total-size=100G --num-threads=4 run | tee memory-test-1-$current.txt

    # Memory-Test-2
    sysbench --test=memory --memory-block-size=1M --memory-total-size=1T --num-threads=4 run | tee memory-test-2-$current.txt

    # File Read
    sysbench --num-threads=16 --test=fileio --file-total-size=5G --file-test-mode=rndrd prepare &&
echo3 > /proc/sys/vm/drop_caches
    sysbench --num-threads=16 --test=fileio --file-total-size=5G --file-test-mode=rndrd run | tee fileio-test-rndrd-$current.txt &&
echo3 > /proc/sys/vm/drop_caches
    sysbench --num-threads=16 --test=fileio --file-total-size=5G --file-test-mode=rndrd cleanup &&
echo3 > /proc/sys/vm/drop_caches


    # File Write
    sysbench --num-threads=16 --test=fileio --file-total-size=5G --file-test-mode=rndwr prepare &&
echo 3 > /proc/sys/vm/drop_caches
    sysbench --num-threads=16 --test=fileio --file-total-size=5G --file-test-mode=rndwr run | tee fileio-test-rndwr-$current.txt &&
echo 3 > /proc/sys/vm/drop_caches
    sysbench --num-threads=16 --test=fileio --file-total-size=5G --file-test-mode=rndwr cleanup &&
echo 3 > /proc/sys/vm/drop_caches

    current=$((current+1))
done
~
```

For each time, we have to exit the current docker image and run the following command to test different cases:

```
$ docker run --cpus=1 --memory=4096M -it a3d53c9bd81f /bin/bash
$ docker run --cpus=4 --memory=4096M -it a3d53c9bd81f /bin/bash
$ docker run --cpus=1 --memory=16384M -it a3d53c9bd81f /bin/bash
$ docker run --cpus=4 --memory=16384M -it a3d53c9bd81f /bin/bash
```

Create a sysbench file folder each time using the following command:

```
$ mkdir sysbench
```

Copy “run-sysbench.sh” from vm to docker using the following command:

```
docker cp run-sysbench.sh e1f2778efde9:/sysbench
```

```
root@fedora:/home/holly# docker ps
CONTAINER ID   IMAGE          COMMAND       CREATED          STATUS          PORTS     NAMES
e1f2778efde9   a3d53c9bd81f   "/bin/bash"   29 seconds ago   Up 27 seconds             strange_carson
root@fedora:/home/holly# docker cp run-sysbench.sh e1f2778efde9:/sysbench
Successfully copied 3.58kB to e1f2778efde9:/sysbench
root@fedora:/home/holly#
```

Be Carefully to copy to the right container ID, each time the container ID will be changed after each docker run

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


Prime numbers limit: 20000

Initializing worker threads...

Threads started!

CPU speed:
    events per second:   121.55

General statistics:
    total time:                          10.0524s
    total number of events:              1222

Latency (ms):
         min:                                    1.92
         avg:                                   32.90
         max:                                   88.55
         95th percentile:                       86.00
         sum:                                40202.82

Threads fairness:
    events (avg/stddev):           305.5000/7.40
    execution time (avg/stddev):   10.0507/0.00
```

```
  block size: 1KiB
  total size: 102400MiB
  operation: write
  scope: global

Initializing worker threads...

Threads started!

Total operations: 9852921 (978687.41 per second)

9621.99 MiB transferred (955.75 MiB/sec)


General statistics:
    total time:                          10.0618s
    total number of events:              9852921

Latency (ms):
         min:                                    0.00
         avg:                                    0.00
         max:                                   77.01
         95th percentile:                        0.00
         sum:                                28561.10

Threads fairness:
    events (avg/stddev):           2463230.2500/138141.90
    execution time (avg/stddev):   7.1403/0.35
```

```
Initializing worker threads...

Threads started!


File operations:
    reads/s:                      2552.93
    writes/s:                     0.00
    fsyncs/s:                     0.00

Throughput:
    read, MiB/s:                  39.89
    written, MiB/s:               0.00

General statistics:
    total time:                          10.0677s
    total number of events:              25717

Latency (ms):
         min:                                    0.00
         avg:                                    6.24
         max:                                  334.02
         95th percentile:                       41.85
         sum:                                160475.13

Threads fairness:
    events (avg/stddev):           1607.3125/85.78
    execution time (avg/stddev):   10.0297/0.02
```

Test Result

Using the following command to get data result

```
$ cat cpu-test-1-*.txt | grep 'events per second' | awk '{printf("%s\t", $4)}'
$ cat cpu-test-2-*.txt | grep 'events per second' | awk '{printf("%s\t", $4)}'
$ cat memory-test-1-*.txt | grep 'Total operations' | awk '{printf("%s\t", $4)}'
$ cat memory-test-2-*.txt | grep 'Total operations' | awk '{printf("%s\t", $4)}'
$ cat fileio-test-rndrd-*.txt | grep 'read, MiB/s:' | awk '{printf("%s\t", $3)}'
$ cat fileio-test-rndwr-*.txt | grep 'written, MiB/s' | awk '{printf("%s\t", $3)}'
```

Result

			Round 1	Round 2	Round 3	Round 4	Round 5	Round 6	Round 7	Round 8	Round 9	Round 10	STD	AVG	MAX	MIN
	CPU Test 1	events per second	121.55	144.83	159.86	201.04	127.51	136.5	143.64	175.45	161.35	154.14	24.9774684	152.4144444	201.04	121.55
	CPU Test 2	events per second	126.6	142.84	164.04	200.27	118.73	148.75	110.63	169.09	193.75	139.18	31.74723457	152.7444444	200.27	110.63
	Memory Test 1	operations per second	974369.02	739986.97	978687.41	840140.21	668883.31	792238.58	774093.1	1127744.39	842794.25	669434.54	142497.6658	859881.9156	1127744.39	668883.31
	Memory Test 2	operations per second	4682.1	7196.58	5551.53	5528.87	5157.06	5764.75	7351.14	5909.48	6696.62	5458.75	913.3581309	5982.014444	7351.14	4682.1
--cpus=1	fileio-test-mdrd-	read, MiB/s	46.15	39.89	42.51	36.06	40.21	40.29	41.33	41.2	39.86	39.11	2.663761438	40.83333333	46.15	36.06
--memory=4096M	fileio-test-mdwr-	written, MiB/s	1.11	2.99	1.61	3.01	2.31	1.33	3.66	1.73	1.31	2.25	0.9130413158	2.117777778	3.66	1.11

			Round 1	Round 2	Round 3	Round 4	Round 5	Round 6	Round 7	Round 8	Round 9	Round 10	STD	AVG	MAX	MIN
	CPU Test 1	events per second	2069.38	2059.46	2062.02	2051.33	2057.14	2063.51	2092.63	2057.63	2088.93	2047.73	14.43857314	2066.892222	2092.63	2051.33
	CPU Test 2	events per second	2081.12	2063.52	2071.69	2088.01	2068.52	2059.51	2084.64	2078.35	2069.95	2047.56	9.686415746	2073.923333	2088.01	2059.51
	Memory Test 1	operations per second	10385260.65	9477928.17	9873553.23	10722670.94	9713046.73	10762396.06	10541667.38	11281107.53	9969504.51	10024735.93	584382.5657	10303015.02	11281107.53	9477928.17
	Memory Test 2	operations per second	77561.09	77248.3	77615.01	77458.35	78022.6	76652.3	77999.53	77468.56	78173.93	74831.83	464.227876	77577.74111	78173.93	76652.3
--cpus=4	fileio-test-mdrd-	read, MiB/s	43.01	42.03	46.49	43.16	45.92	42.79	45.47	47.04	43.01	39.22	1.882565478	44.32444444	47.04	42.03
--memory=4096M	fileio-test-mdwr-	written, MiB/s	3.18	3.39	1.92	4.28	4.08	3.29	3.1	1.31	1.52	3.21	1.072392186	2.896666667	4.28	1.31

			Round 1	Round 2	Round 3	Round 4	Round 5	Round 6	Round 7	Round 8	Round 9	Round 10	STD	AVG	MAX	MIN
	CPU Test 1	events per second	153.06	283.37	143.3	176.25	126.6	190.47	169.34	152.58	114.74	120.03	49.37872875	167.7455556	283.37	114.74
	CPU Test 2	events per second	213.99	190.43	150.11	152.69	142.16	180.18	149.88	234.19	114.17	113.78	38.02642637	169.7555556	234.19	114.17
	Memory Test 1	operations per second	825196.53	1019266.34	977112.34	989144.17	973010.04	602732.03	918490.08	550589.55	642526.02	675872.45	185700.0228	833118.5667	1019266.34	550589.55
	Memory Test 2	operations per second	5784.67	5950.01	5502.81	5624.34	4453.96	4274.36	5059.21	4784.91	4171.61	4495.58	678.3038695	5067.32	5950.01	4171.61
--cpus=1	fileio-test-mdrd-	read, MiB/s	1008.05	873.63	1048.52	1029.37	772.85	1016.78	885.19	865.83	720.67	820.83	118.7918401	913.4322222	1048.52	720.67
--memory=16384M	fileio-test-mdwr-	written, MiB/s	1.44	2.79	1.71	2.5	1.58	2.23	1.67	3.11	1.61	2.8	0.6071129311	2.071111111	3.11	1.44

			Round 1	Round 2	Round 3	Round 4	Round 5	Round 6	Round 7	Round 8	Round 9	Round 10	STD	AVG	MAX	MIN
	CPU Test 1	events per second	2063.25	2070.58	2050.39	2103.51	2089.79	2059.92	2064.06	2060.35	2067.36	2073.9	16.51487192	2069.912222	2103.51	2050.39
	CPU Test 2	events per second	2073.98	2060.46	2082.96	2101.18	2059.85	2073.16	2094.36	2075.09	2099.06	2096.57	15.49925115	2080.011111	2101.18	2059.85
	Memory Test 1	operations per second	11469933.97	10908309.03	10612743.13	10328730.34	11145893.67	10694276.75	10594834.01	10642183.63	10682224.83	10798710.36	340545.374	10786569.93	11469933.97	10328730.34
	Memory Test 2	operations per second	79576.12	79593.71	77893.42	77456.48	77942.04	77075.29	79305.3	77895.15	79223.5	78952.86	978.3756947	78440.11222	79593.71	77075.29
--cpus=4	fileio-test-mdrd-	read, MiB/s	2708.45	2876.19	3123.23	2721.03	2917.17	2842.04	2895.07	2745.18	3032.43	2746.28	140.5114271	2873.421111	3123.23	2708.45
--memory=16384M	fileio-test-mdwr-	written, MiB/s	1.3	2.06	3.06	1.46	3.47	1.64	3.18	2.38	2.58	1.77	0.7892999711	2.347777778	3.47	1.3

Result Analysis

	--cpus=1 --memory=4096M	--cpus=4 --memory=4096M	--cpus=1 --memory=16384M	--cpus=4 --memory=16384M
CPU Test-1 --test=cpu --cpu-max-prime=2000 --num-threads=4 run	152.4144444	2066.892222	167.7455556	2069.912222
CPU Test-2 --test=cpu --cpu-max-prime=20000 --num-threads=4 run	152.7444444	2073.923333	169.7555556	2080.011111
Memory Test-1 --test=memory --memory-block-size=1K --memory-total-size=100G --num-threads=4 run	859881.9156	10303015.02	833118.5667	10786569.93
Memory Test-2 --test=memory --memory-block-size=1M --memory-total-size=1T --num-threads=4 run	5982.014444	77577.74111	5067.32	78440.11222
FileIO Test-1 --num-threads=16 --test=fileio --file-total-size=5G --file-test-mode=rndrd	40.83333333	44.32444444	913.4322222	2873.421111

FileIO Test-2 --num-threads=16 --test=fileio --file-total-size=5G --file-test-mode=rndwr	2.117777778	2.896666667	2.071111111	2.347777778

The above data was the average of 10 times for each case.
From the table:

1. cpu-max-prime=2000 is similar to cpu-max-prime=20000, both of them have similar events per second.
2. memory-block-size=1K --memory-total-size=100G has 100 times operations per second than memory-block-size=1M --memory-total-size=1T
3. For all test cases, FileIO written, MiB/s is similar.
4. All the data in CPUs=4 is 4 times than CPUs=4
5. RAM 4096 and RAM 16384 are similar to each other, except file-test-mode=rndwr

Therefore, CPUs and memory-block-size affects the performance.

In conclusion, CPUs and memory-block-size affects the performance in both QEMU and Docker. When CPU = 1, QEMU is better than Docker, when CPU=4, Docker is better than QEMU.