Gabinete, Keith Ginoel S. 2020-03670
CMSC 125 – CD7L Nov. 21, 2023

ICS-OS Lab 01: Building and Booting ICS-OS

Task 1: Install Docker and Docker-Compose

I. Install Docker Engine on Ubuntu. <https://docs.docker.com/engine/install/ubuntu/>

a. Uninstall all conflicting packages before installing docker (if there's any).

```
shio@ksgabinete: ~
shio@ksgabinete:~$ for pkg in docker.io docker-doc docker-compose docker-compose-v2 podman-docker
 containerd runc; do sudo apt-get remove $pkg; done
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Package 'docker.io' is not installed, so not removed
0 upgraded, 0 newly installed, 0 to remove and 3 not upgraded.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Package 'docker-doc' is not installed, so not removed
0 upgraded, 0 newly installed, 0 to remove and 3 not upgraded.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Package 'docker-compose' is not installed, so not removed
0 upgraded, 0 newly installed, 0 to remove and 3 not upgraded.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Package 'docker-compose-v2' is not installed, so not removed
0 upgraded, 0 newly installed, 0 to remove and 3 not upgraded.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Package 'podman-docker' is not installed, so not removed
0 upgraded, 0 newly installed, 0 to remove and 3 not upgraded.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Package 'containerd' is not installed, so not removed
0 upgraded, 0 newly installed, 0 to remove and 3 not upgraded.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Package 'runc' is not installed, so not removed
0 upgraded, 0 newly installed, 0 to remove and 3 not upgraded.
shio@ksgabinete:~$
```

b. Set up Docker's apt repository.

```
4:34 AM
shio@ksgabinete: ~
shio@ksgabinete:~$ # Add Docker's official GPG key:
sudo apt-get update
sudo apt-get install ca-certificates curl gnupg
sudo install -m 0755 -d /etc/apt/keyrings
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/doc
ker.gpg
sudo chmod a+r /etc/apt/keyrings/docker.gpg

# Add the repository to Apt sources:
echo \
  "deb [arch="$(dpkg --print-architecture)" signed-by=/etc/apt/keyrings/docker.gpg] https://download.
docker.com/linux/ubuntu \
  "$(. /etc/os-release && echo "$VERSION_CODENAME")" stable" | \
  sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
sudo apt-get update
Hit:1 http://ph.archive.ubuntu.com/ubuntu jammy InRelease
Hit:2 http://ph.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:3 http://ph.archive.ubuntu.com/ubuntu jammy-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu jammy-security InRelease
Hit:5 https://dl.winehq.org/wine-builds/ubuntu jammy InRelease
Hit:6 https://ppa.launchpadcontent.net/yannubuntu/boot-repair/ubuntu jammy InRelease
Reading package lists... Done
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
ca-certificates is already the newest version (20230311ubuntu0.22.04.1).
ca-certificates set to manually installed.
gnupg is already the newest version (2.2.27-3ubuntu2.1).
gnupg set to manually installed.
The following NEW packages will be installed:
  curl
0 upgraded, 1 newly installed, 0 to remove and 3 not upgraded.
Need to get 194 kB of archives.
After this operation, 454 kB of additional disk space will be used.
Do you want to continue? [Y/n] Y
Get:1 http://ph.archive.ubuntu.com/ubuntu jammy-updates/main amd64 curl amd64 7.81.0-1ubuntu1.14 [194
 kB]
Fetched 194 kB in 1s (361 kB/s)
Selecting previously unselected package curl.
(Reading database ... 212403 files and directories currently installed.)
Preparing to unpack .../curl_7.81.0-1ubuntu1.14_amd64.deb ...
Unpacking curl (7.81.0-1ubuntu1.14) ...
Setting up curl (7.81.0-1ubuntu1.14) ...
Processing triggers for man-db (2.10.2-1) ...
Hit:1 http://ph.archive.ubuntu.com/ubuntu jammy InRelease
Hit:2 http://ph.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:3 http://ph.archive.ubuntu.com/ubuntu jammy-backports InRelease
Get:4 https://download.docker.com/linux/ubuntu jammy InRelease [48.8 kB]
Hit:5 http://security.ubuntu.com/ubuntu jammy-security InRelease
Get:6 https://download.docker.com/linux/ubuntu jammy/stable amd64 Packages [22.7 kB]
Hit:7 https://ppa.launchpadcontent.net/yannubuntu/boot-repair/ubuntu jammy InRelease
Hit:8 https://dl.winehq.org/wine-builds/ubuntu jammy InRelease
Fetched 71.5 kB in 1s (53.9 kB/s)
Reading package lists... Done
shio@ksgabinete:~$
```

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

```
4:38 AM
shio@ksgabinete: ~
Created symlink /etc/systemd/system/sockets.target.wants/docker.socket → /lib/systemd/system/docker.s
ocket.
Processing triggers for man-db (2.10.2-1) ...
shio@ksgabinete:~$ sudo docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
719385e32844: Pull complete
Digest: sha256:c79d06dfdfd3d3eb04cafd0dc2bacab0992ebc243e083cabe208bac4dd7759e0
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/

shio@ksgabinete:~$ sudo docker run hello-world

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/

shio@ksgabinete:~$
```

II. Install the Docker's Compose plugin. <https://docs.docker.com/compose/install/linux/#install-using-the-repository>

a. Set up the repository. (already did in the previous section)

b. Update the package index, and install the latest version of Docker Compose.

```
4:41 AM
shio@ksgabinete: ~
shio@ksgabinete:~$ sudo apt-get update
Hit:1 http://ph.archive.ubuntu.com/ubuntu jammy InRelease
Hit:2 http://ph.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:3 http://ph.archive.ubuntu.com/ubuntu jammy-backports InRelease
Hit:4 https://download.docker.com/linux/ubuntu jammy InRelease
Hit:5 http://security.ubuntu.com/ubuntu jammy-security InRelease
Hit:6 https://dl.winehq.org/wine-builds/ubuntu jammy InRelease
Hit:7 https://ppa.launchpadcontent.net/yannubuntu/boot-repair/ubuntu jammy InRelease
Reading package lists... Done
shio@ksgabinete:~$ sudo apt-get install docker-compose-plugin
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
docker-compose-plugin is already the newest version (2.21.0-1~ubuntu.22.04~jammy).
0 upgraded, 0 newly installed, 0 to remove and 3 not upgraded.
```

c. Verify that Docker Compose is installed correctly by checking the version.

```
shio@ksgabinete:~$ docker compose version
Docker Compose version v2.21.0
shio@ksgabinete:~$
```

III. Linux post-installation steps for Docker Engine (Manage Docker as a non-root user)

a. Create the docker group.

```
4:43 AM
shio@ksgabinete: ~
shio@ksgabinete:~$ sudo groupadd docker
groupadd: group 'docker' already exists
shio@ksgabinete:~$
```

b. Add your user to the docker group.

```
shio@ksgabinete:~$ sudo usermod -aG docker $USER
shio@ksgabinete:~$
```

c. Log out and log back in so that your group membership is re-evaluated. (did off Cam)

activate the changes to groups:

```
shio@ksgabinete: ~
shio@ksgabinete:~$ newgrp docker
```

d. Verify that you can run docker commands without sudo.

```
shio@ksgabinete:~$ docker run hello-world

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/

shio@ksgabinete:~$
```

Additional: Configure Docker to start on boot with system

```
shio@ksgabinete:~$ sudo systemctl enable docker.service
Synchronizing state of docker.service with SysV service script with /lib/systemd/systemd-sysv-install
.
Executing: /lib/systemd/systemd-sysv-install enable docker
shio@ksgabinete:~$ sudo systemctl enable containerd.service
shio@ksgabinete:~$
```

Task 2: Clone the repository and explore the source tree

I. Checkout the source code and explore the source tree. (clone https://github.com/srg-ics-uplb/ics-os.git into a local directory)

```
shio@ksgabinete: ~
shio@ksgabinete:~$ git clone https://github.com/srg-ics-uplb/ics-os.git ics-os-ksgabinete
Cloning into 'ics-os-ksgabinete'...
remote: Enumerating objects: 3547, done.
remote: Counting objects: 100% (398/398), done.
remote: Compressing objects: 100% (188/188), done.
remote: Total 3547 (delta 212), reused 396 (delta 210), pack-reused 3149
Receiving objects: 100% (3547/3547), 25.76 MiB | 503.00 KiB/s, done.
Resolving deltas: 100% (2364/2364), done.
```

Note: ‘ics-os-ksgabinete’ was renamed to ‘ics-os-kgsg’ (can be seen in the succeeding screenshots)

II. Create a branch for lab01 to make code management easier.

```
shio@ksgabinete:~$ cd ics-os-kgsg/
shio@ksgabinete:~/ics-os-kgsg$ git checkout -b lab01
Switched to a new branch 'lab01'
shio@ksgabinete:~/ics-os-kgsg$ git branch
* lab01
  master
shio@ksgabinete:~/ics-os-kgsg$
```

Task 3: Build ICS-OS kernel

I. Open a new terminal. Start and enter the container.

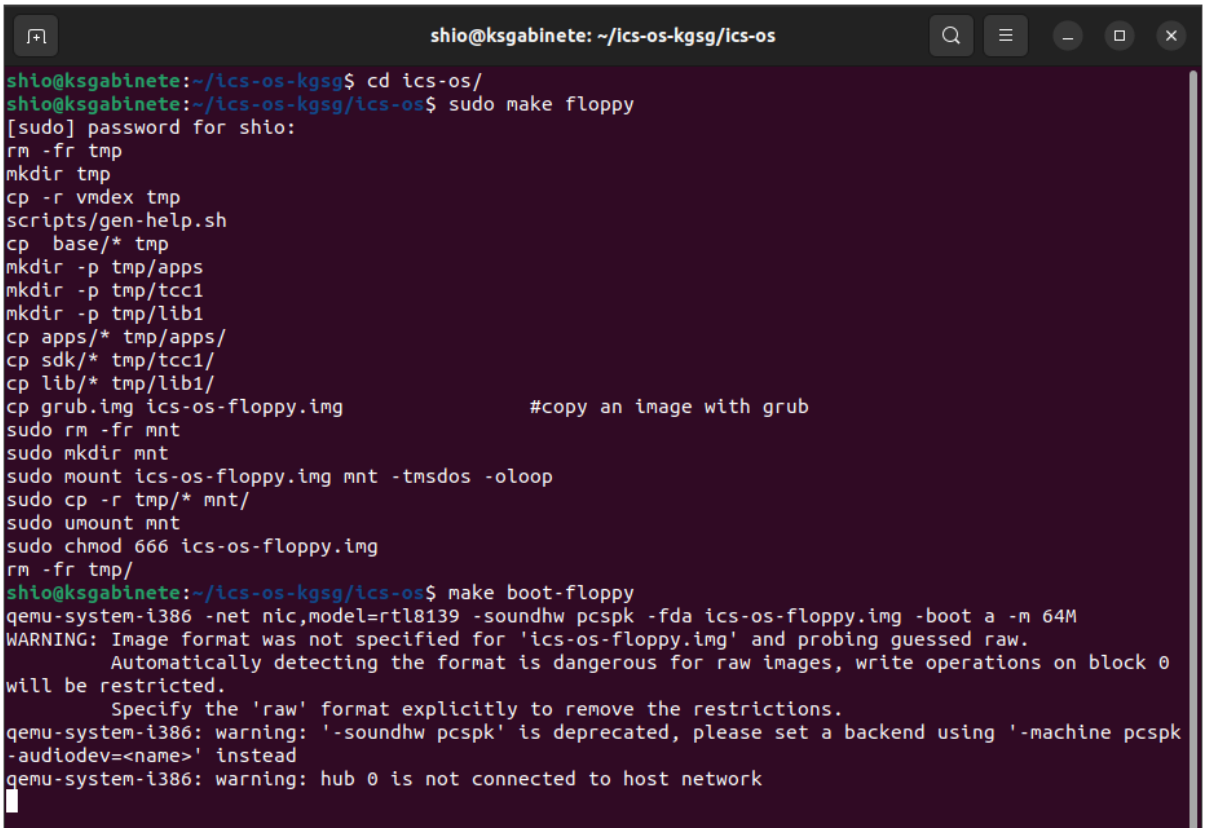
```
shio@ksgabinete:~/ics-os-kgsg/ics-os$ docker compose run ics-os-build
[+] Creating 1/1
 ✔ Network ics-os_default  Created                                                    0.2s
[+] Building 91.1s (8/8) FINISHED                                          docker:default
 => [ics-os-build internal] load build definition from Dockerfile                     0.0s
 => => transferring dockerfile: 328B                                                  0.0s
 => [ics-os-build internal] load .dockerignore                                        0.0s
 => => transferring context: 2B                                                       0.0s
 => [ics-os-build internal] load metadata for docker.io/library/ubuntu:16.04          6.1s
 => [ics-os-build 1/4] FROM docker.io/library/ubuntu:16.04@sha256:1f1a2d56de1d604801a9671f301  11.7s
 => => resolve docker.io/library/ubuntu:16.04@sha256:1f1a2d56de1d604801a9671f301190704c25d604a  0.0s
 => => sha256:1f1a2d56de1d604801a9671f301190704c25d604a416f59e03c04f5c6ffee0d6 1.42kB / 1.42kB  0.0s
 => => sha256:a3785f78ab8547ae2710c89e627783cfa7ee7824d3468cae6835c9f4eae23ff7 1.15kB / 1.15kB  0.0s
 => => sha256:b6f50765242581c887ff1acc2511fa2d885c52d8fb3ac8c4bba131fd86567f2e 3.36kB / 3.36kB  0.0s
 => => sha256:58690f9b18fca6469a14da4e212c96849469f9b1be6661d2342a4bf01774a 46.50MB / 46.50MB  10.4s
 => => sha256:b51569e7c50720acf6860327847fe342a1afbe148d24c529fb81df105e3eed01 857B / 857B      0.8s
 => => sha256:da8ef40b9ecabc2679fe2419957220c0272a965c5cf7e0269fa1aeeb8c56f2e1 528B / 528B      0.6s
 => => sha256:fb15d46c38dcd1ea0b1990006c3366ecd10c79d374f341687eb2cb23a2c8672e 170B / 170B      1.2s
 => => extracting sha256:58690f9b18fca6469a14da4e212c96849469f9b1be6661d2342a4bf01774aa50       1.1s
 => => extracting sha256:b51569e7c50720acf6860327847fe342a1afbe148d24c529fb81df105e3eed01       0.0s
 => => extracting sha256:da8ef40b9ecabc2679fe2419957220c0272a965c5cf7e0269fa1aeeb8c56f2e1       0.0s
 => => extracting sha256:fb15d46c38dcd1ea0b1990006c3366ecd10c79d374f341687eb2cb23a2c8672e       0.0s
 => [ics-os-build 2/4] RUN apt-get update                                             8.0s
 => [ics-os-build 3/4] RUN apt-get install -y build-essential nasm qemu-kvm tcc git gcc-multi  62.9s
 => [ics-os-build 4/4] RUN mkdir -p /home/ics-os                                      0.5s
 => [ics-os-build] exporting to image                                                 1.8s
 => => exporting layers                                                               1.8s
 => => writing image sha256:1e3b2b9de76dc92ca8d1929a7864aa67c72b595d047755197f9e011d470a8118    0.0s
 => => naming to docker.io/library/ics-os-ics-os-build                                0.0s
root@b2a85ef0597e:/#
```

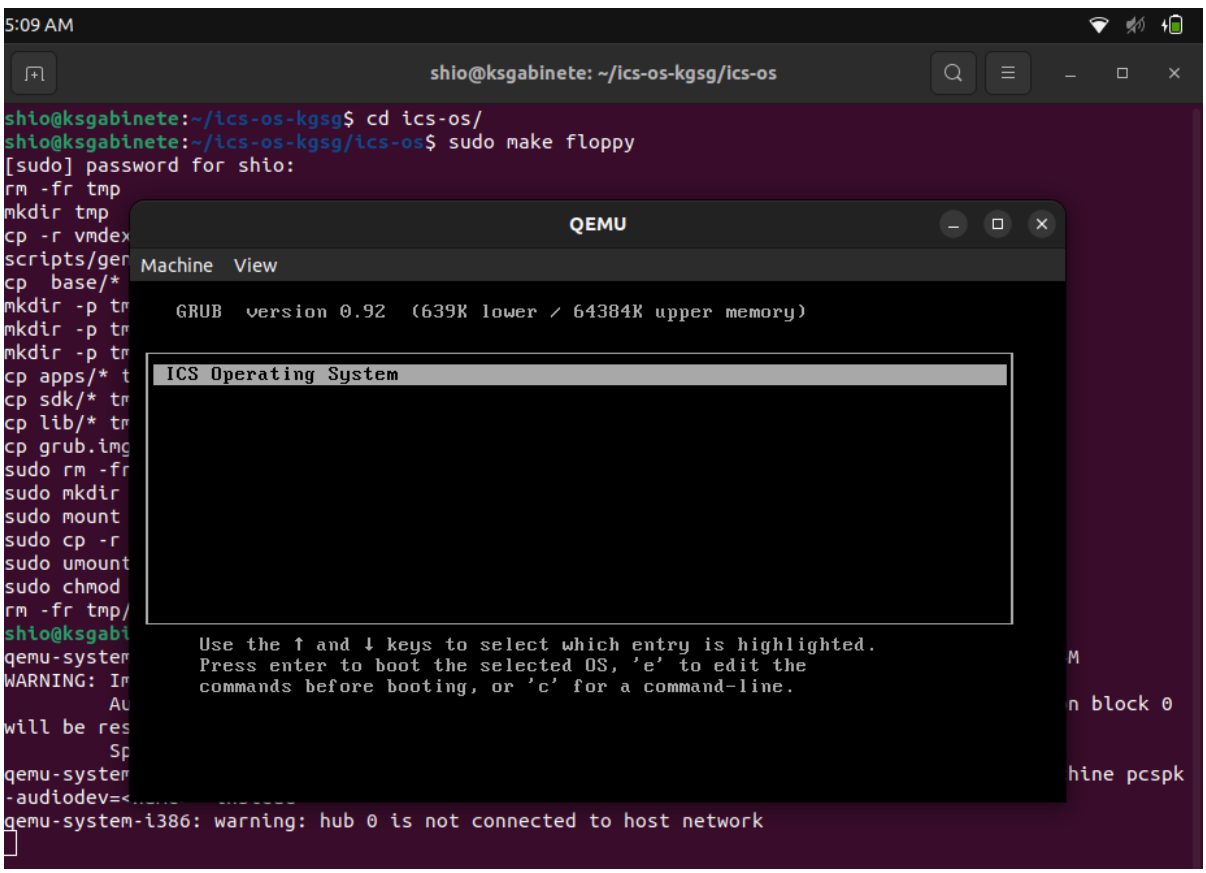
I. Build the kernel image.

```
root@b2a85ef0597e:/# /#cd /home/ics-os
bash: /#cd: No such file or directory
root@b2a85ef0597e:/# cd /home/ics-os
root@b2a85ef0597e:/home/ics-os# make clean
rm -f vmdex
rm -f ics-os-livecd.iso
rm -fr tmp/*
make -C kernel/ clean
make[1]: Entering directory '/home/ics-os/kernel'
rm -f *.o
rm -f Kernel32.bin
rm -f Kernel32.sym
rm -f vmdex
make[1]: Leaving directory '/home/ics-os/kernel'
root@b2a85ef0597e:/home/ics-os# make
make -C kernel/
make[1]: Entering directory '/home/ics-os/kernel'
gcc -fno-stack-protector -fgnu89-inline -m32 -w -nostdlib -fno-builtin -ffreestanding -c -g  -o sched
uler.o process/scheduler.c
gcc -fno-stack-protector -fgnu89-inline -m32 -w -nostdlib -fno-builtin -ffreestanding -c -g  -o fat.o
 filesystem/fat12.c
gcc -fno-stack-protector -fgnu89-inline -m32 -w -nostdlib -fno-builtin -ffreestanding -c -g  -o iso96
60.o filesystem/iso9660.c
gcc -fno-stack-protector -fgnu89-inline -m32 -w -nostdlib -fno-builtin -ffreestanding -c -g  -o devfs
.o filesystem/devfs.c
gcc -fno-stack-protector -fgnu89-inline -m32 -w -nostdlib -fno-builtin -ffreestanding -c -g  -o iomgr
.o iomgr/iosched.c
gcc -fno-stack-protector -fgnu89-inline -m32 -w -nostdlib -fno-builtin -ffreestanding -c -g  -o devmg
r_error.o devmgr/devmgr_error.c
gcc -fno-stack-protector -fgnu89-inline -m32 -w -nostdlib -fno-builtin -ffreestanding -c -g  -o kerne
l32.o kernel32.c
nasm -f elf32 -o startup.o startup/startup.asm
nasm -f elf32 -o asmlib.o startup/asmlib.asm
startup/asmlib.asm:321: warning: label alone on a line without a colon might be in error
nasm -f elf32 -o irqwrap.o irqwrap.asm
#strip --strip-debug *.o
ld -melf_i386  -T lscript.ld -Map mapfile.txt
objcopy --only-keep-debug Kernel32.bin Kernel32.sym
objcopy --strip-debug Kernel32.bin
gzip -c -9 Kernel32.bin >  vmdex
cp vmdex ..
make[1]: Leaving directory '/home/ics-os/kernel'
root@b2a85ef0597e:/home/ics-os#
```

Task 4: Create the disk and boot ICS-OS

I. Build the boot floppy then start Qemu with the floppy image as boot device.





Task 5: Run ICS-OS commands

I. Once the ICS-OS command prompt (%) appears, type help. Examine the list of commands and run two of these commands.

I. typing “help”

```
QEMU
Machine  View
Institute of Computer Science
University of the Philippines, Los Banos

Type "help" on the command prompt to
display available commands.

/icsos/ %help
Name: /icsos/icsos.hlp  Size: 2232 bytes
ICS-OS Commands
---------------
add- Adds two integers. Args: <num1> <num2>
cc- Builds a C program (invokes tcc.exe). Args: <name.exe> <name.c>
cd- Changes working directory. Args: <directory>
cls- Clears the screen.
copy- Copy source to destination: Args: <source> <destination>
cpuid- Displays CPU information.
del- Deletes a files or directory. Args: <filename/dirname>
demo_graphics- Runs the graphics demonstration.
dkill- Dirty kill a user process/thread. No cleanup is done. Args: <pid>
echo- Displays a string. Args: <string>
exit- Exits a console session.
fgman-  Foreground manager

Press any key to continue, 'q' to quit
```

II. running two commands

a. “cpuid"

```
Machine  View
/icsos/ %cpuid
Manufacturer     : GenuineIntel
Product String   : QEMU Virtual CPU version 2.5+
CPU model        : Model 6, Family 6, Type 0, Stepping 3.
Features     :
* FPU
* Debugging Extensions
* Page Size Extensions
* Time stamp counter
* Model-specific registers
* Physical Address Extensions
* Machine Check Exceptions
* Compare and exchange 8-byte
* On-chip APIC
* iFSC
* PGE
* CMOV
* Page Attribute Table
* MMX
* Fast floating point save/restore
* Intel SSE
* Intel SSE2

/icsos/ %_
```

b. “lsdev”

```
Machine  View
/icsos/ %lsdev
ID    Device Name     Type            Description
1     stdlib          Kernel Service  Standard Library Interface
2     mem_mgr         memory manager  DEX low-level memory manager
3     bsd_malloc      Kernel Extension University of California, Berkeley malloc

4     dl_malloc       Kernel Extension Doug Lea's dlmalloc VERSION 2.7.2
5     dex_malloc      Kernel Extension malloc for debugging
6     hwportmgr       port manager    Hardware port manager
7     keyb            character device Default generic keyboard driver 1.00
8     mouse           character device Default mouse driver 1.00
9     default_sched   Kernel Extension Default Round-Robin Scheduler
10    fd0             Block device    Generic Floppy Disk Controller driver
11    cds0            Block device    ATA s master CD/DVD-ROM (READ-ONLY)
12    vga             Unknown         VGA graphics driver
13    default_iomgr   I/O manager     DEX default I/O scheduler and manager
14    null            Block device    null block device
15    devfs           Filesystem      Device Filesystem
16    fat             Filesystem      MS-DOS FATfs driver
17    cdfs            Filesystem      DEX ISO 9660/Joliet CD-ROM filesystem
18    ramdisk         Block device    Virtual Block Device
/icsos/ %
```

Task 6: Cleanup

I. Exit the build container by typing the command “exit”.

```
QEMU - Press Ctrl+Alt+G to release grab
Machine  View
Warning: 6. Cannot resolve 'Sleep'
Warning: 7. Cannot resolve 'WaitForSingleObject'
Warning: 0. Cannot resolve '__dllonexit'
Warning: 2. Cannot resolve 'abort'
Number of Data directories: 16
examining export directory..
obtaining DEX specific entrypoints..
Number of exports - 1
fxn names address = 0xe0016038
fxn addresses     = 0xe0016034
function name: dex32_libmain
Calling dex32 compatible library entrypoint located at e000d420..
DEX Ramdisk Driver 1.03
10000 blocks alloated
Identifying FAT type..
Identifying FAT type..
Using FAT16..
allocating FAT..
writing to FAT..formatting..
done.
Installing Ramdisk as ramdisk
Initialization successful!
Driver was assigned handle 18
Call successful.
_
```

II. Go back to the master branch of the source code.

```
shio@ksgabinete: ~/ics-os-kgsg/ics-os
shio@ksgabinete:~/ics-os-kgsg$ cd ics-os/
shio@ksgabinete:~/ics-os-kgsg/ics-os$ sudo make floppy
[sudo] password for shio:
rm -fr tmp
mkdir tmp
cp -r vmdex tmp
scripts/gen-help.sh
cp  base/* tmp
mkdir -p tmp/apps
mkdir -p tmp/tcc1
mkdir -p tmp/lib1
cp apps/* tmp/apps/
cp sdk/* tmp/tcc1/
cp lib/* tmp/lib1/
cp grub.img ics-os-floppy.img                #copy an image with grub
sudo rm -fr mnt
sudo mkdir mnt
sudo mount ics-os-floppy.img mnt -tmsdos -oloop
sudo cp -r tmp/* mnt/
sudo umount mnt
sudo chmod 666 ics-os-floppy.img
rm -fr tmp/
shio@ksgabinete:~/ics-os-kgsg/ics-os$ make boot-floppy
qemu-system-i386 -net nic,model=rtl8139 -soundhw pcspk -fda ics-os-floppy.img -boot a -m 64M
WARNING: Image format was not specified for 'ics-os-floppy.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.
qemu-system-i386: warning: '-soundhw pcspk' is deprecated, please set a backend using '-machine pcspk
-audiodev=<name>' instead
qemu-system-i386: warning: hub 0 is not connected to host network
shio@ksgabinete:~/ics-os-kgsg/ics-os$ git checkout master
M       ics-os/ics-os-floppy.img
Switched to branch 'master'
Your branch is up to date with 'origin/master'.
shio@ksgabinete:~/ics-os-kgsg/ics-os$ git branch
  lab01
* master
shio@ksgabinete:~/ics-os-kgsg/ics-os$
```

Reflection: Write some realizations and questions that crossed your mind while doing this lab.

While doing this lab, I came to realize that ICS-OS is also a Linux distribution (distro), similar to Ubuntu. I noticed that most, if not all, of the commands in ICS-OS are based on Linux built-in commands, named and functioning exactly like their Linux counterparts. One question that comes to mind, though, is why do we need to use Docker to run the ICS operating system? What are containers exactly for? And can we run ICS-OS without these containers?

A few other questions that crossed my mind while doing this lab are: “Can I install the ICS operating system as a stand-alone package in an empty disk drive?” “Wouldn’t I encounter problems running the ICS-OS on my device due to some missing drivers or driver incompatibility?” “Do I have to write drivers for each specific component that makes up my device, or can I use the already-existing drivers from other operating systems?”

Obviously, I still have much to learn about how operating systems work and are built. So, for now, I believe further research on my part is necessary.