

Container Technology with Docker

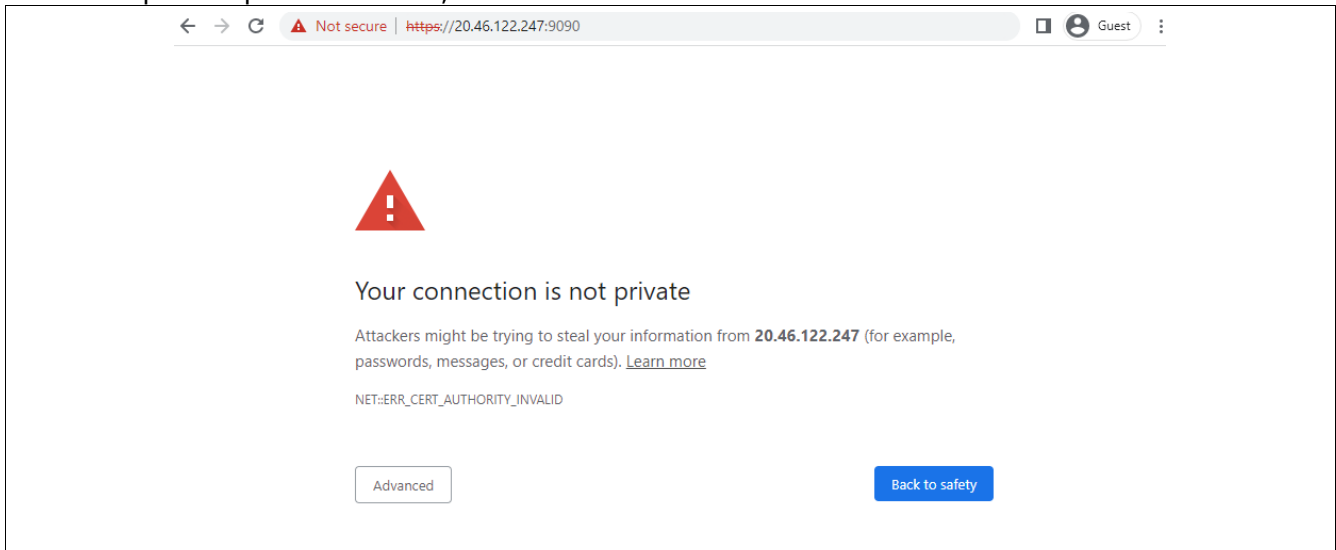
LAB SHEET

Lab Note:

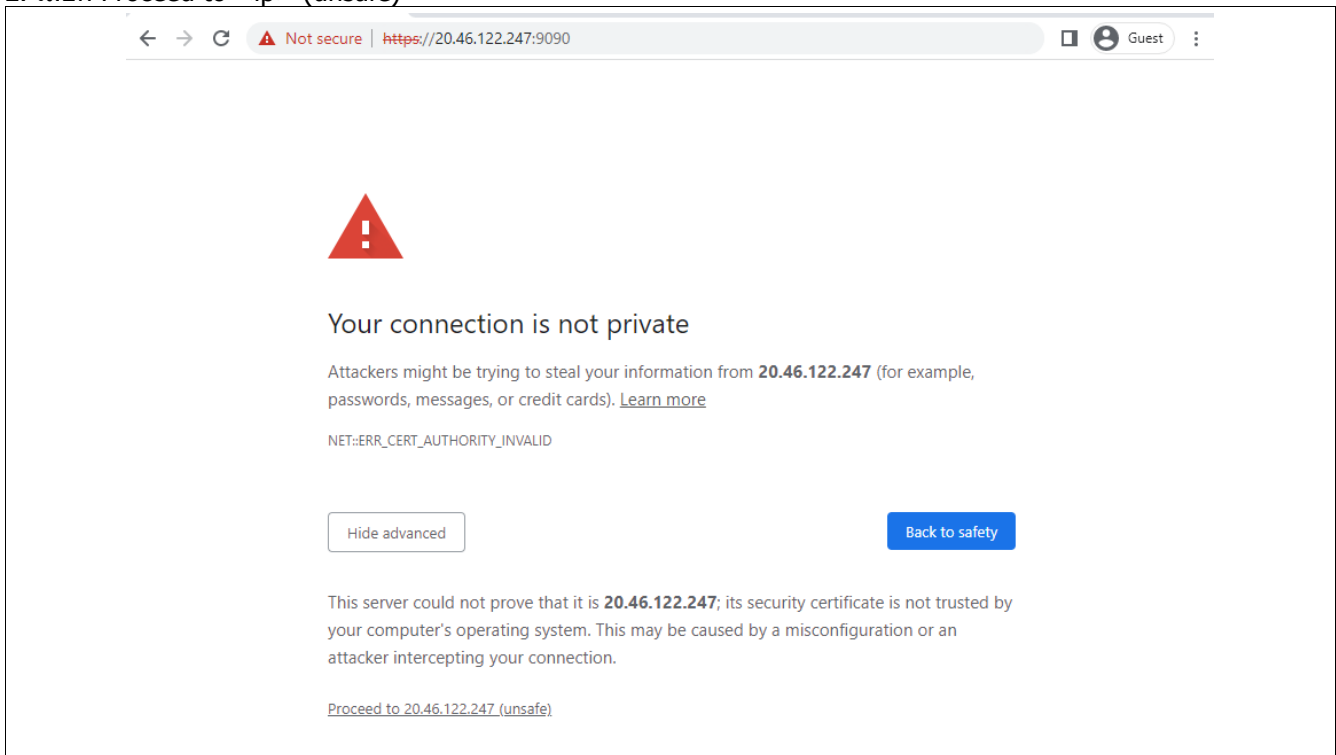
User Information		
IP	Training-01	20.191.184.12
	Training-02	20.63.185.64
	Training-03	20.191.190.208
	Training-04	20.194.168.108
	Training-05	20.191.190.98
	Training-06	20.191.191.114
Username	student	
Password	dX22#training	

0. Installation Lab

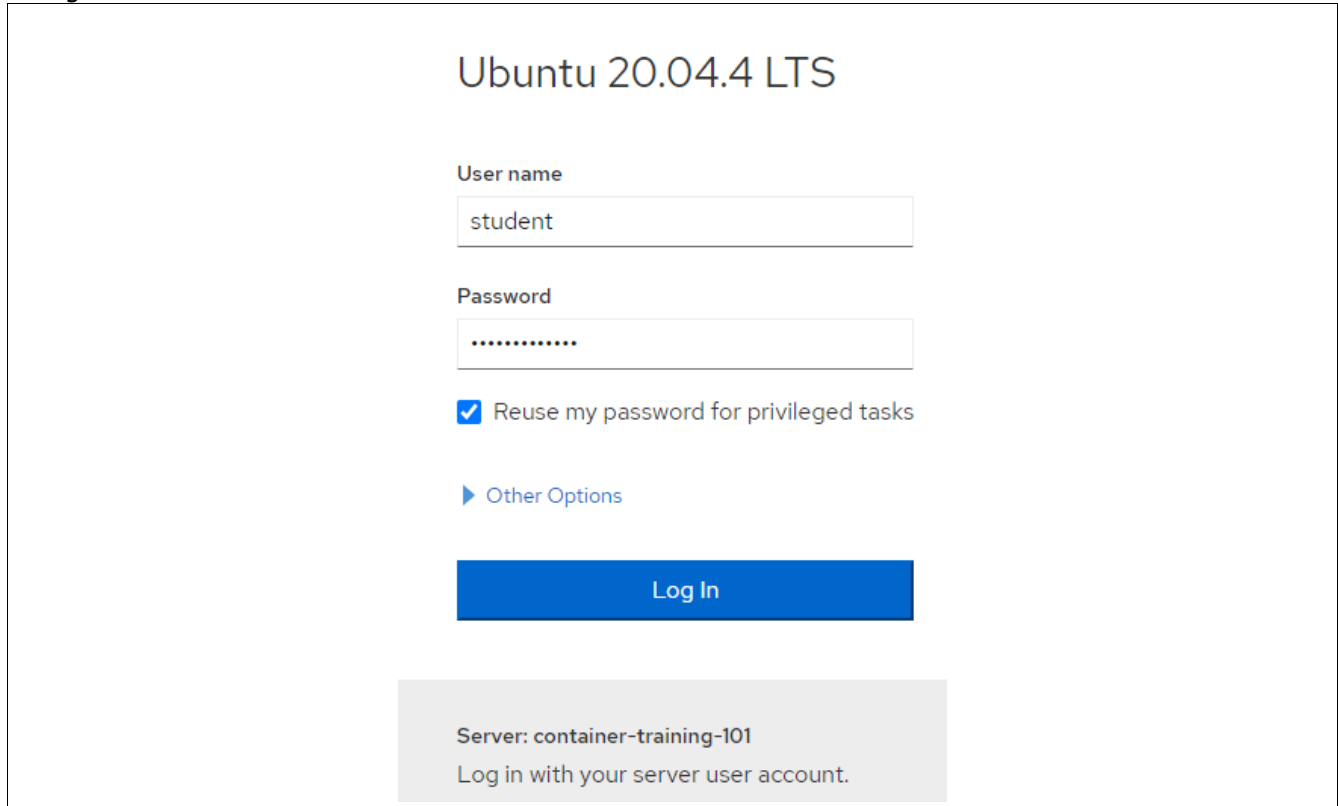
1. ทำการ login โดยเปิด Chrome browser และ browse ไปที่ <https://<ip>:9090> แทนที่ <ip> ด้วย ip ตามลำดับเครื่อง, เลือก Advanced



2. เลือก Proceed to <ip> (unsafe)



3. Log In ด้วย User name และ Password ที่กำหนด



Ubuntu 20.04.4 LTS

User name

Password

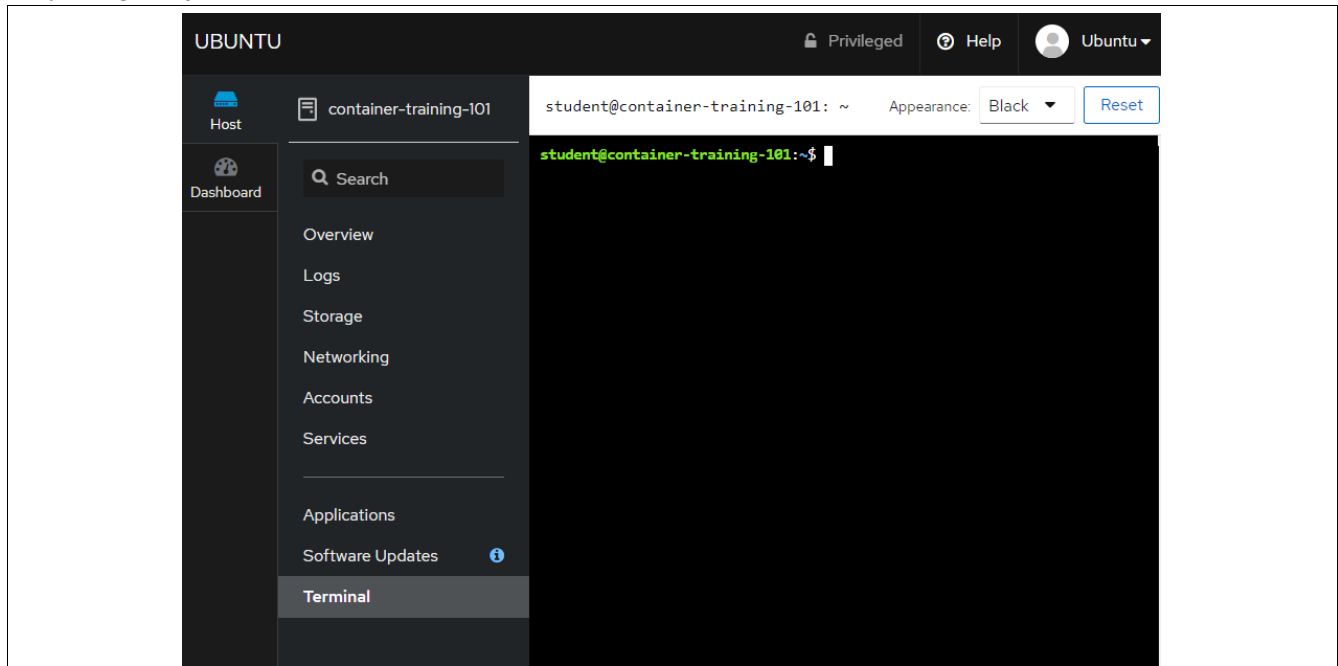
☒ Reuse my password for privileged tasks

[▶ Other Options](#)

[Log In](#)

Server: container-training-101
Log in with your server user account.

4. เลือก Terminal



5. อัปเดต apt package index และติดตั้ง packages เพื่อใช้งาน repository ผ่าน HTTPS

```
sudo apt update
sudo apt-get install ca-certificates curl gnupg lsb-release
```

```
student@container-training-X:~$ sudo apt update
...
student@container-training-X:~$ sudo apt-get install ca-certificates curl gnupg lsb-release
l gnupg lsb-release
Reading package lists... Done
Building dependency tree
Reading state information... Done
lsb-release is already the newest version (11.1.0ubuntu2).
ca-certificates is already the newest version (20211016~20.04.1).
curl is already the newest version (7.68.0-1ubuntu2.12).
gnupg is already the newest version (2.2.19-3ubuntu2.2).
0 upgraded, 0 newly installed, 0 to remove and 12 not upgraded.
student@container-training-X:~$

student@container-training-X:~$ sudo mkdir -p /etc/apt/keyrings
student@container-training-X:~$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o
/etc/apt/keyrings/docker.gpg
student@container-training-X:~$
```

6. เพิ่ม Docker Official GPG Key

```
sudo mkdir -p /etc/apt/keyrings
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg
```

```
student@container-training-X:~$ sudo mkdir -p /etc/apt/keyrings
student@container-training-X:~$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o
/etc/apt/keyrings/docker.gpg
student@container-training-X:~$
```

7. เพิ่ม Repository

```
echo "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg]
https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable" | sudo tee
/etc/apt/sources.list.d/docker.list > /dev/null
```

```
student@container-training-X:~$ echo "deb [arch=$(dpkg --print-architecture) signed-
by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/ubuntu $(lsb_release -cs)
stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
student@container-training-X:~$
```

8. อัปเดต apt package index และติดตั้ง docker engine

```
sudo apt-get update
sudo apt-get install docker-ce docker-ce-cli containerd.io docker-compose-plugin
```

พิมพ์ Y เพื่อตกลงติดตั้ง

```
student@container-training-X:~$ sudo apt-get update
...
student@container-training-X:~$ sudo apt-get install docker-ce docker-ce-cli containerd.io docker-
compose-plugin
...
Setting up docker-compose-plugin (2.6.0~ubuntu-focal) ...
Setting up docker-ce-cli (5:20.10.17~3-0~ubuntu-focal) ...
Setting up pigz (2.4-1) ...
Setting up docker-ce-rootless-extras (5:20.10.17~3-0~ubuntu-focal) ...
Setting up docker-ce (5:20.10.17~3-0~ubuntu-focal) ...
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /lib/systemd/system/docker.service.
Created symlink /etc/systemd/system/sockets.target.wants/docker.socket → /lib/systemd/system/docker.socket.
Processing triggers for man-db (2.9.1-1) ...
Processing triggers for systemd (245.4-4ubuntu3.17) ...
student@container-training-101:~$
```


9. สั่งให้ docker เริ่มทำงาน และ กำหนดให้ docker ทำงานในขั้นตอนการ start ระบบ docker

```
sudo systemctl enable --now docker
Sudo systemctl status docker
```

```
student@container-training-X:~$ sudo systemctl enable --now docker
Synchronizing state of docker.service with SysV service script with /lib/s
ystemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable docker
student@container-training-X:~$ sudo systemctl status docker
● docker.service - Docker Application Container Engine
   Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor >
   Active: active (running) since Wed 2022-08-31 03:07:12 UTC; 3min 31s>
 TriggeredBy: ● docker.socket
   Docs: https://docs.docker.com
  Main PID: 16331 (dockerd)
    Tasks: 8
   Memory: 34.0M
   CGroup: /system.slice/docker.service
           └─16331 /usr/bin/dockerd -H fd:// --containerd=/run/containe>

Aug 31 03:07:11 container-training-101 dockerd[16331]: time="2022-08-31T0>
Aug 31 03:07:11 container-training-101 dockerd[16331]: time="2022-08-31T0>
Aug 31 03:07:11 container-training-101 dockerd[16331]: time="2022-08-31T0>
Aug 31 03:07:12 container-training-101 dockerd[16331]: time="2022-08-31T0>
Aug 31 03:07:12 container-training-101 dockerd[16331]: time="2022-08-31T0>
Aug 31 03:07:12 container-training-101 dockerd[16331]: time="2022-08-31T0>
Aug 31 03:07:12 container-training-101 dockerd[16331]: time="2022-08-31T0>
Aug 31 03:07:12 container-training-101 dockerd[16331]: time="2022-08-31T0>
Aug 31 03:07:12 container-training-101 systemd[1]: Started Docker Applica>
Aug 31 03:07:12 container-training-101 dockerd[16331]: time="2022-08-31T0>
lines 1-21/21 (END)
```

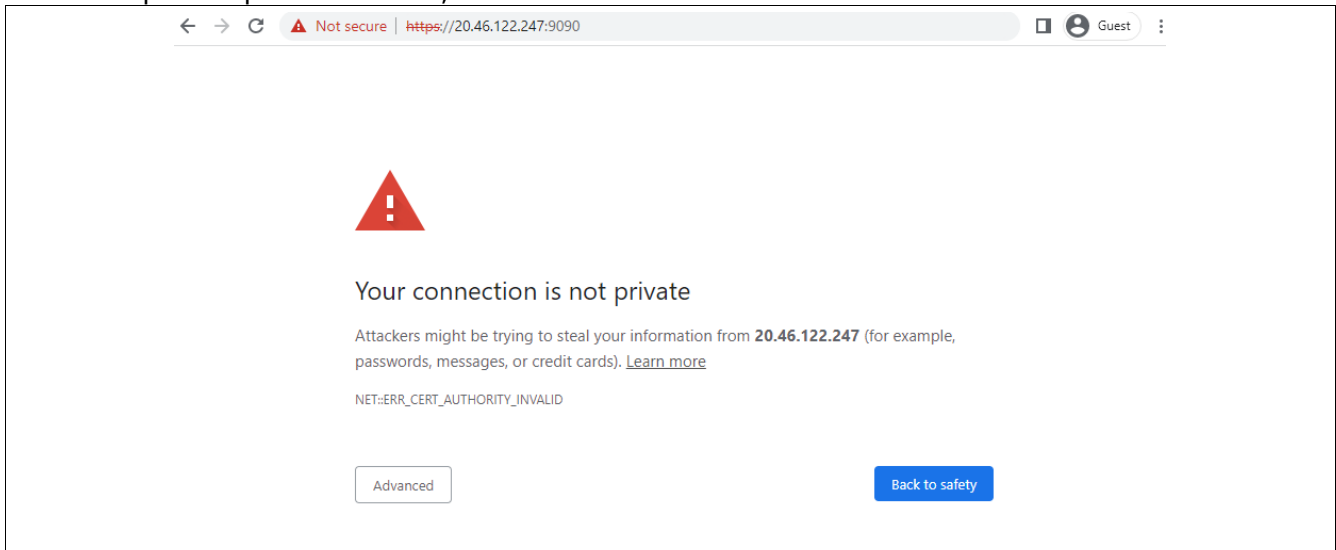
10. ตรวจสอบความถูกต้องการติดตั้ง docker

```
student@container-training-X:~$ sudo docker --version
Docker version 20.10.17, build 100c701
student@container-training-X:~$ sudo docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
2db29710123e: Pull complete
Digest: sha256:7d246653d0511db2a6b2e0436cfd0e52ac8c066000264b3ce63331ac66d
ca625
Status: Downloaded newer image for hello-world:latest

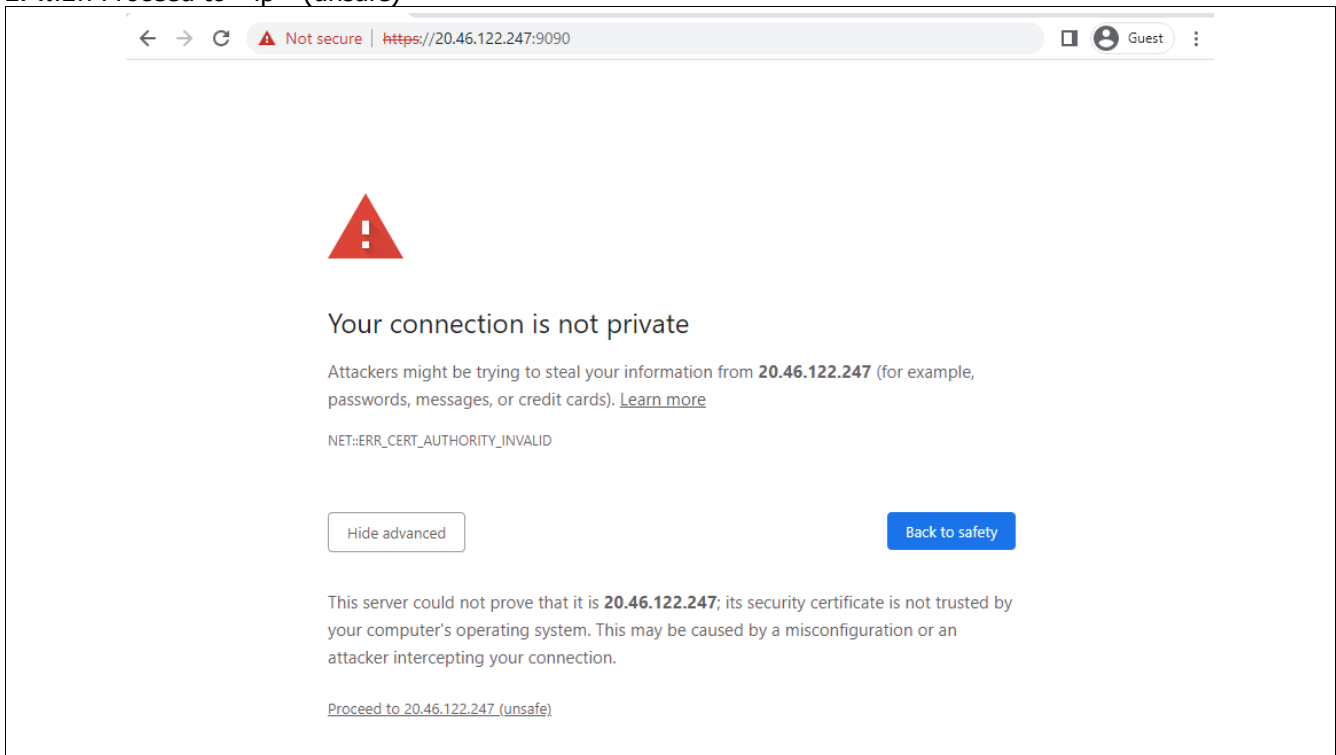
Hello from Docker!
This message shows that your installation appears to be working correctly.
```

1. Docker Commands

1. ทำการ login โดยเปิด Chrome browser และ browse ไปที่ <https://<ip>:9090> แทนที่ <ip> ด้วย ip ตามลำดับเครื่อง, เลือก Advanced



2. เลือก Proceed to <ip> (unsafe)



3. Log In ด้วย User name และ Password ที่กำหนด

Ubuntu 20.04.4 LTS

User name

Password

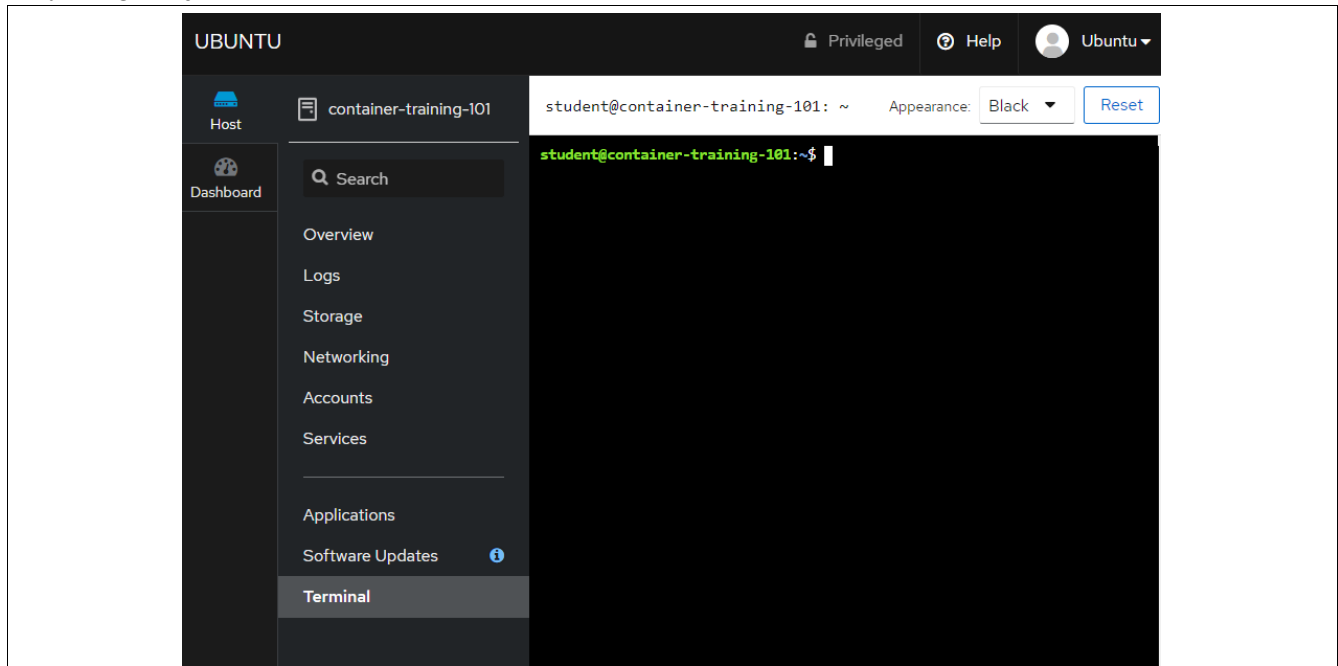
☒ Reuse my password for privileged tasks

[▶ Other Options](#)

[Log In](#)

Server: container-training-101
Log in with your server user account.

4. เลือก Terminal



5. ใช้คำสั่ง docker pull เพื่อ pull image ชื่อ centos จาก dockerhub

```
student@container-training-X:~$ sudo docker pull centos
Using default tag: latest
latest: Pulling from library/centos
a1d0c7532777: Pull complete
Digest: sha256:a27fd8080b517143cbbbab9dfb7c8571c40d67d534bbdee55bd6c473f432b177
Status: Downloaded newer image for centos:latest
docker.io/library/centos:latest
student@container-training-X:~$
```

6. Run container จาก image centos ที่ได้ pull ไว้ก่อนหน้านี้ โดยให้ทำงานแบบ detached และตั้งชื่อ mycentos1

```
student@container-training-101:~$ sudo docker run -d --name mycentos1 centos
b2dc25233802332d9f327fbd5484ed15f47cc9dc08adcacc9eeeb26ca09a541c
student@container-training-101:~$ sudo docker container ls
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS   NAMES
student@container-training-101:~$ sudo docker container ls -a
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS   NAMES
b2dc25233802   centos    "/bin/bash" 11 seconds ago Exited (0) 10 seconds ago   mycentos1
4638fa07bdc0   hello-world "/hello"   51 minutes ago Exited (0) 51 minutes ago   condescending_robinson
student@container-training-101:~$
```

Note. เหตุใด Container ที่เพิ่งสร้างจาก image centos จึงอยู่ในสถานะ Exited

7. Run container อีกครั้ง โดยครั้งนี้เพิ่มคำสั่งให้ทำ sleep 120 และตั้งชื่อ mycentos2

```
student@container-training-101:~$ sudo docker run -d --name mycentos2 centos sleep 120
b9a7ce697e5162b4a6db956572f65e5ed7d91f5aef265c1dc157f000a6f3f65d
student@container-training-101:~$ sudo docker container ls
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS        PORTS   NAMES
b9a7ce697e51   centos    "sleep 120"             7 seconds ago Up 6 seconds   mycentos2
student@container-training-101:~$
```

8. Run container แบบ Foreground และตั้งชื่อ mycentos3

```
student@container-training-101:~$ sudo docker run --name mycentos3 centos sleep 20
```

9. Run container แบบ Foreground โดยเพิ่มการ attach interactive ตั้งชื่อ mycentos4

```
student@container-training-101:~$ sudo docker run --name mycentos4 -it centos /bin/bash
[root@979dcf7c1791 /]# cat /etc/os-release
NAME="CentOS Linux"
VERSION="8"
ID="centos"
ID_LIKE="rhel fedora"
VERSION_ID="8"
PLATFORM_ID="platform:el8"
PRETTY_NAME="CentOS Linux 8"
ANSI_COLOR="0;31"
CPE_NAME="cpe:/o:centos:centos:8"
HOME_URL="https://centos.org/"
BUG_REPORT_URL="https://bugs.centos.org/"
CENTOS_MANTISBT_PROJECT="CentOS-8"
CENTOS_MANTISBT_PROJECT_VERSION="8"
[root@979dcf7c1791 /]#
```

10. Run container ชื่อ mycentos5 จะได้ UUID แบบยาว จากนั้นเรียกดู UUID แบบสั้น ทำการ inspect เพื่อตรวจสอบข้อมูลจาก UUID แบบสั้น โดยสามารถระบุ format ให้แสดงเฉพาะ Id หรือ Name ได้ *เมื่อรันคำสั่ง docker inspect ให้ใช้ UUID แบบสั้นที่ได้จากการรันบนเครื่องนั้นๆ

```
student@container-training-101:~$ sudo docker run --name mycentos5 -d centos sleep 600
6e331fa5eaf2a7a521e0b52a44b2005fd77b80a89d8ea97648d9c8c1025675ab
student@container-training-101:~$ sudo docker container ls
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS          PORTS          NAMES
6e331fa5eaf2   centos    "sleep 600"             5 seconds ago   Up 4 seconds           mycentos5
student@container-training-101:~$ sudo docker inspect 6e331fa5eaf2
[
  {
    "Id": "6e331fa5eaf2a7a521e0b52a44b2005fd77b80a89d8ea97648d9c8c1025675ab",
    ...

student@container-training-101:~$ sudo docker inspect -f "{{ .ID }}" 6e331fa5eaf2
6e331fa5eaf2a7a521e0b52a44b2005fd77b80a89d8ea97648d9c8c1025675ab
student@container-training-101:~$ sudo docker inspect -f "{{ .Name }}" 6e331fa5eaf2
/mycentos5
```

11. Run container ชื่อ mycentos6 โดยกำหนดให้เขียน Container id ไปยังไฟล์ชื่อ mycid

```
student@container-training-101:~$ cat mycid
cat: mycid: No such file or directory
student@container-training-101:~$ sudo docker run --name mycentos6 -d --cidfile "mycid" centos
a83df2076baa74775cf2aa4f0b112ac1d80a8dd930522ae955ae9ec7ecbc1624
student@container-training-101:~$ cat mycid
a83df2076baa74775cf2aa4f0b112ac1d80a8dd930522ae955ae9ec7ecbc1624student@container-training-101:~$
```

12. สร้าง container ชื่อ mycentos7 โดยที่ยังไม่เริ่มการทำงาน จากนั้นจึงสั่งให้ container mycentos7 เริ่มทำงาน

```
student@container-training-101:~$ sudo docker create --name mycentos7 -it centos
7853a9fd0522a6d04deab065a84dd99047d9f45dd261456193f0d72492e7f5a4
student@container-training-101:~$ sudo docker ps -a | grep mycentos7
7853a9fd0522 centos "/bin/bash" 5 seconds ago Created mycentos7

student@container-training-101:~$ sudo docker start mycentos7
student@container-training-101:~$ sudo docker ps -a | grep mycentos7
7853a9fd0522 centos "/bin/bash" 17 seconds ago Up 2 seconds mycentos7
```

13. สั่งให้ container mycentos7 หยุดทำงาน

```
student@container-training-101:~$ sudo docker stop mycentos7
mycentos7
student@container-training-101:~$ sudo docker ps -a | grep mycentos7
7853a9fd0522  centos      "/bin/bash"  About a minute ago  Exited (0) 1 second ago      mycentos7
```

14. สั่งให้ container mycentos7 หยุดทำงานและเริ่มทำงานใหม่อีกครั้ง

```
student@container-training-101:~$ sudo docker restart mycentos7
mycentos7
student@container-training-101:~$ sudo docker ps -a | grep mycentos7
7853a9fd0522  centos      "/bin/bash"  2 minutes ago      Up 2 seconds      mycentos7
```


15. Run container ชื่อ mycentos8 จากนั้น Run container ชื่อ mycentos9 แบบกำหนดให้ลบ Container ทันทีที่ทำงานเสร็จ

```
student@container-training-101:~$ sudo docker run --name mycentos8 alpine
student@container-training-101:~$ sudo docker container ls -a | grep mycentos8
4a6ca11f84dc alpine "/bin/sh" 55 seconds ago Exited (0) 54 seconds ago mycentos8

student@container-training-101:~$ sudo docker container run --name mycentos9 --rm alpine
student@container-training-101:~$ sudo docker container ls -a | grep mycentos9
student@container-training-101:~$

*mycentos8 ยังคงอยู่แม้จะ Exited แล้ว
*mycentos9 ถูกลบทันทีที่ทำงานเสร็จ จึงไม่เห็น container นี้
```

16. Run container โดยเพิ่มคำสั่งให้แสดงผล Process id

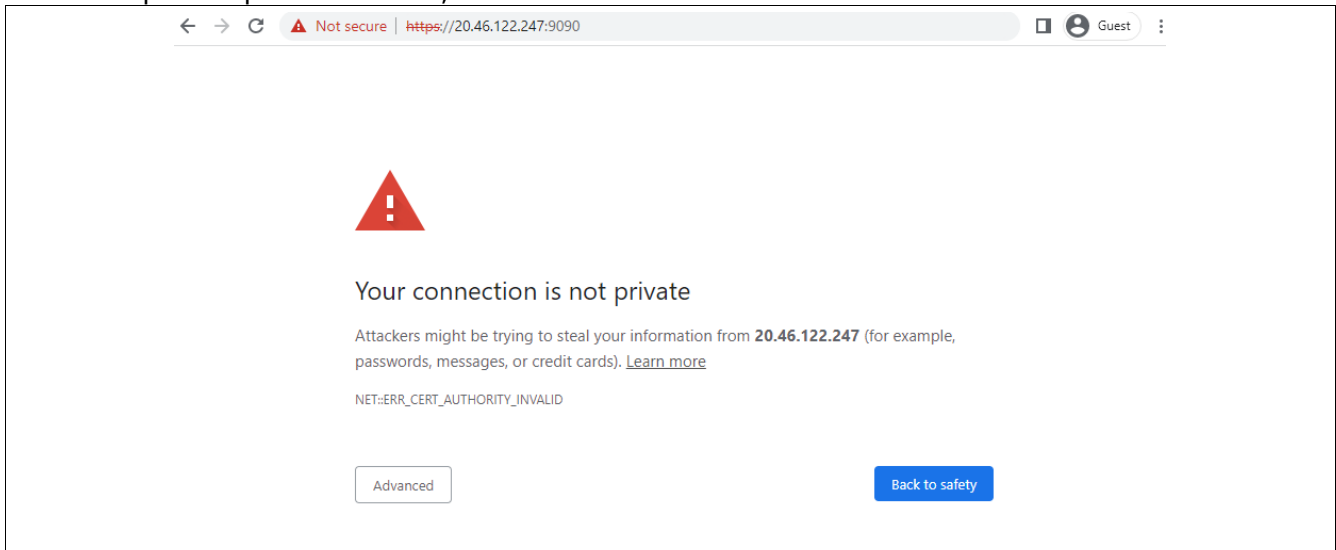
เมื่อเพิ่ม argument `--pid host` จะเห็น Process id ของ docker host

```
student@container-training-101:~$ sudo docker run centos ps aux
USER      PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.0  0.0  42544  2220 ?        Rs    07:50   0:00 ps aux
student@container-training-101:~$ sudo docker run --pid host centos ps aux
...
root      23367  0.0  0.0   9416  4628 ?        S+    07:50   0:00 sudo docker run --pid host centos ps aux
root      23368  2.0  0.6 1273700 51716 ?        Sl+   07:50   0:00 docker run --pid host centos ps aux
...

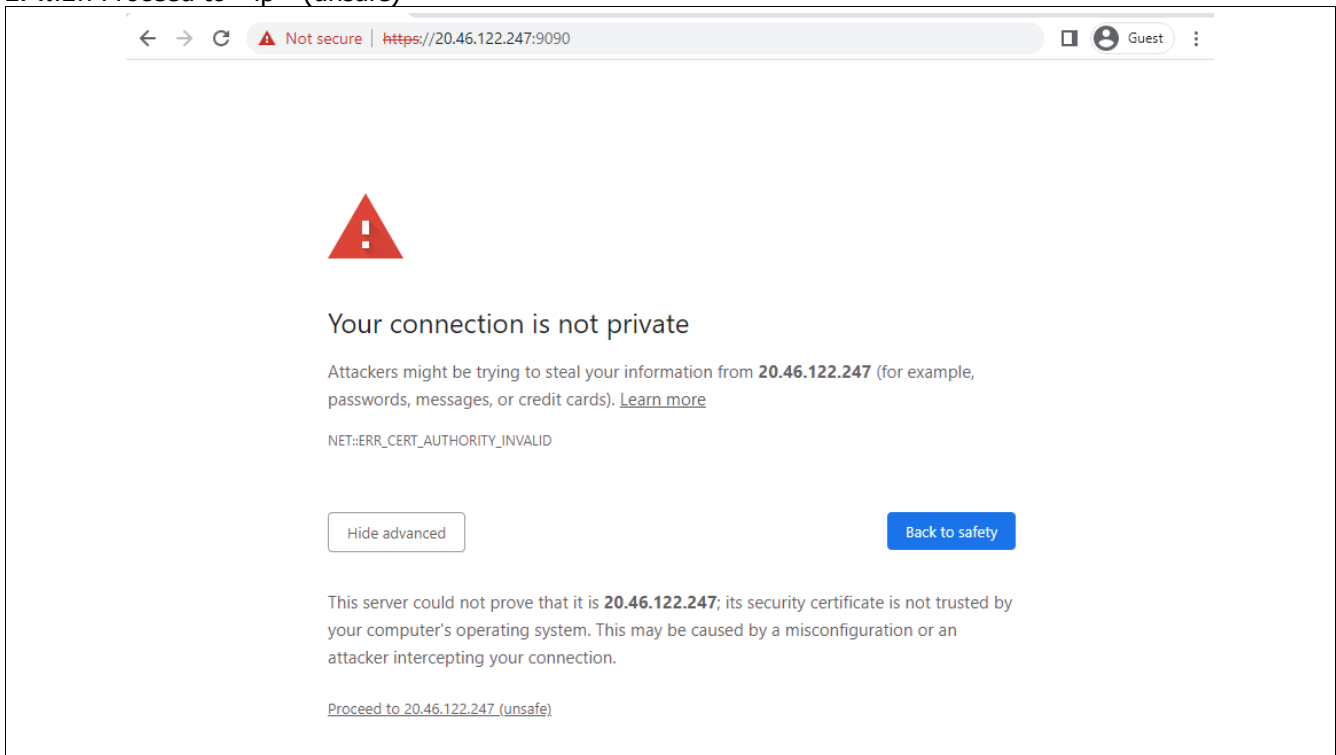
19589 root      0:00 runc --root /var/run/docker/runtime-runc/moby --log
/run/containerd/io.containerd.runtime.v1.linux/moby/cbb68a0eab1ea71fa58b705f3446eff6df0228426ce6513ee81ac9a964d9
--log-format json state cbb68a0eab1ea71fa58b705f3446eff6df0228426ce6513ee81ac9a964d91077
```

2. Docker image and Docker registry

1. ทำการ login โดยเปิด Chrome browser และ browse ไปที่ <https://<ip>:9090>
แทนที่ <ip> ด้วย ip ตามลำดับเครื่อง, เลือก Advanced



2. เลือก Proceed to <ip> (unsafe)



3. Log In ด้วย User name และ Password ที่กำหนด

Ubuntu 20.04.4 LTS

User name

Password

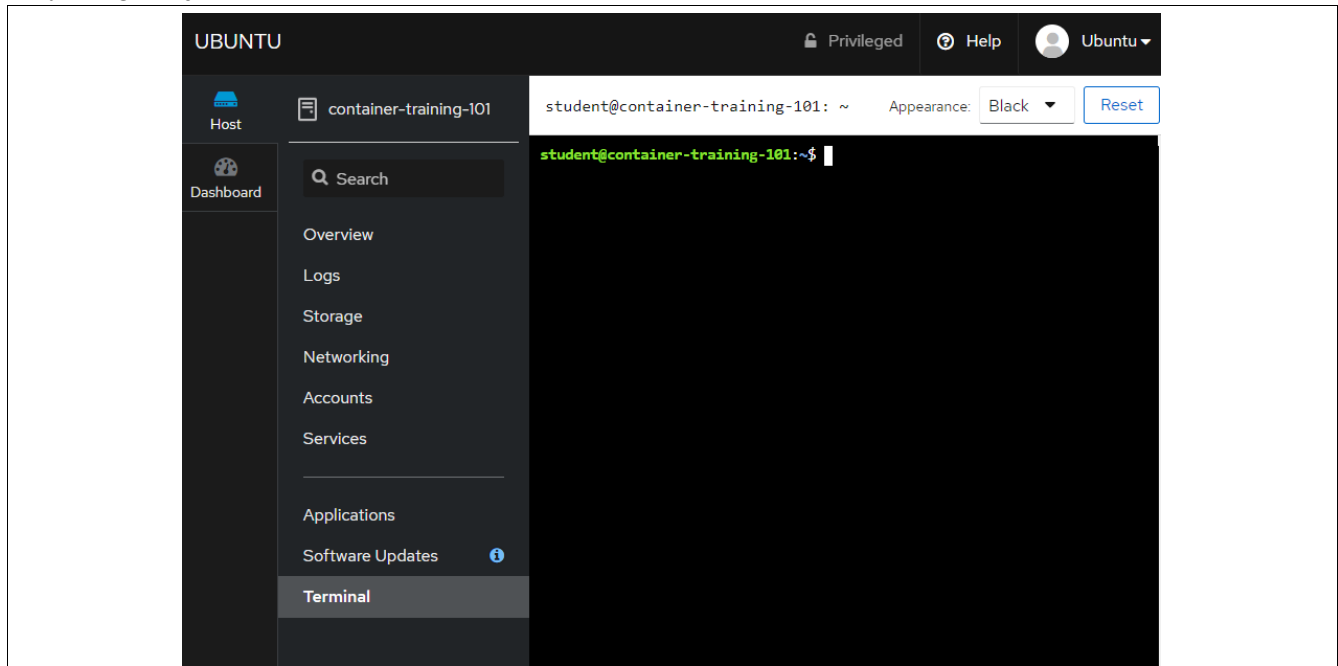
☒ Reuse my password for privileged tasks

[▶ Other Options](#)

[Log In](#)

Server: container-training-101
Log in with your server user account.

4. เลือก Terminal



5. ใช้คำสั่ง docker pull เพื่อ pull image ชื่อ alpine จาก registry dockerhub โดย pull แบบไม่ระบุ tag และแบบระบุ tag 3.5

```
student@container-training-101:~$ sudo docker pull alpine
Using default tag: latest
latest: Pulling from library/alpine
213ec9aee27d: Pull complete
Digest: sha256:bc41182d7ef5ffc53a40b044e725193bc10142a1243f395ee852a8d9730fc2ad
Status: Downloaded newer image for alpine:latest
docker.io/library/alpine:latest
student@container-training-101:~$ sudo docker pull alpine:3.5
3.5: Pulling from library/alpine
8cae0e1ac61c: Pull complete
Digest: sha256:66952b313e51c3bd1987d7c4ddf5dba9bc0fb6e524eed2448fa660246b3e76ec
Status: Downloaded newer image for alpine:3.5
docker.io/library/alpine:3.5
student@container-training-101:~$
```

6. ใช้คำสั่ง docker images เพื่อดู images ที่มีอยู่

```
student@container-training-101:~$ sudo docker images
REPOSITORY      TAG        IMAGE ID      CREATED       SIZE
alpine           latest     9c6f07244728  3 weeks ago   5.54MB
hello-world      latest     feb5d9fea6a5  11 months ago 13.3kB
centos           latest     5d0da3dc9764  11 months ago 231MB
alpine           3.5       f80194ae2e0c  3 years ago   3.99MB
student@container-training-101:~$
```

7. ใช้คำสั่ง docker images โดยระบุ format การแสดงผลเฉพาะ ID, Repository และ Size

```
student@container-training-101:~$ sudo docker images --format "table
{{.ID}}\t{{.Repository}}\t{{.Size}}"
IMAGE ID      REPOSITORY    SIZE
9c6f07244728  alpine        5.54MB
feb5d9fea6a5  hello-world    13.3kB
5d0da3dc9764  centos        231MB
f80194ae2e0c  alpine        3.99MB
student@container-training-101:~$
```

8. ใช้คำสั่ง docker images โดยระบุให้แสดงค่า digests ของแต่ละ image

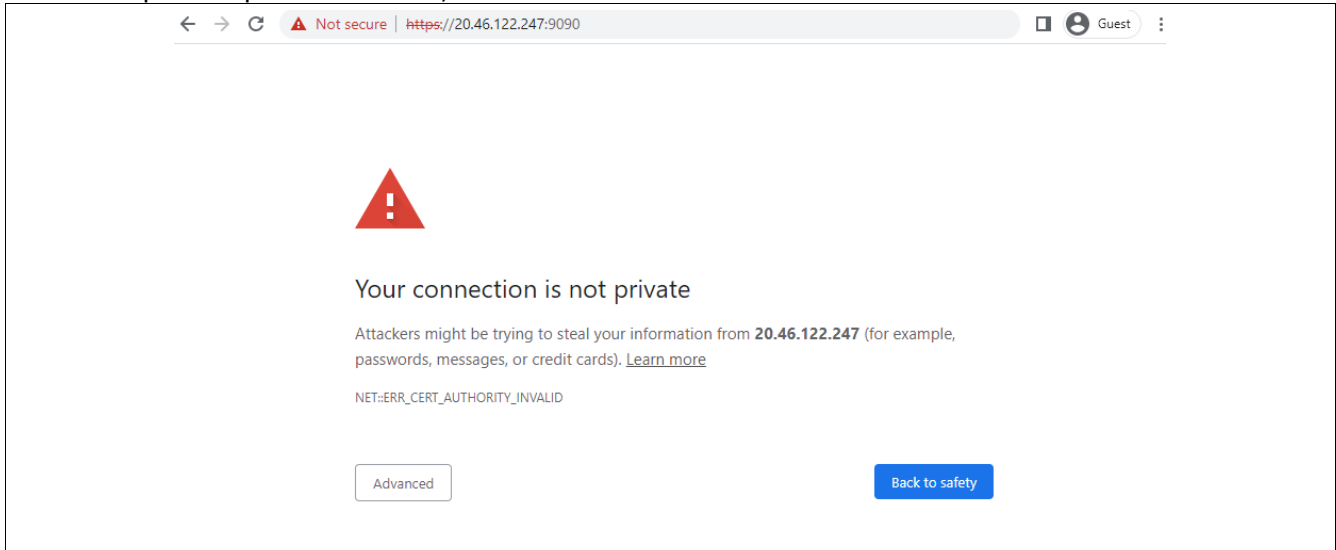
```
student@container-training-101:~$ sudo docker images --digests
REPOSITORY TAG DIGEST IMAGE ID CREATED SIZE
alpine latest sha256:bc41182d7ef5ffc53a40b044e725193bc10142a1243f395ee852a8d9730fc2ad 9c6f07244728 3 weeks ago 5.54MB
hello-world latest sha256:7d246653d0511db2a6b2e0436cfd0e52ac8c066000264b3ce63331ac66dca625 feb5d9fea6a5 11 months ago 13.3kB
centos latest sha256:a27fd8080b517143cbbb9dfb7c8571c40d67d534bbdee55bd6c473f432b177 5d0da3dc9764 11 months ago 231MB
alpine 3.5 sha256:66952b313e51c3bd1987d7c4ddf5dba9bc0fb6e524eed2448fa660246b3e76ec f80194ae2e0c 3 years ago 3.99MB
student@container-training-101:~$
```

9 Run container ชื่อ myalpine1 โดยใช้ค่าจาก digests แทนชื่อ centos

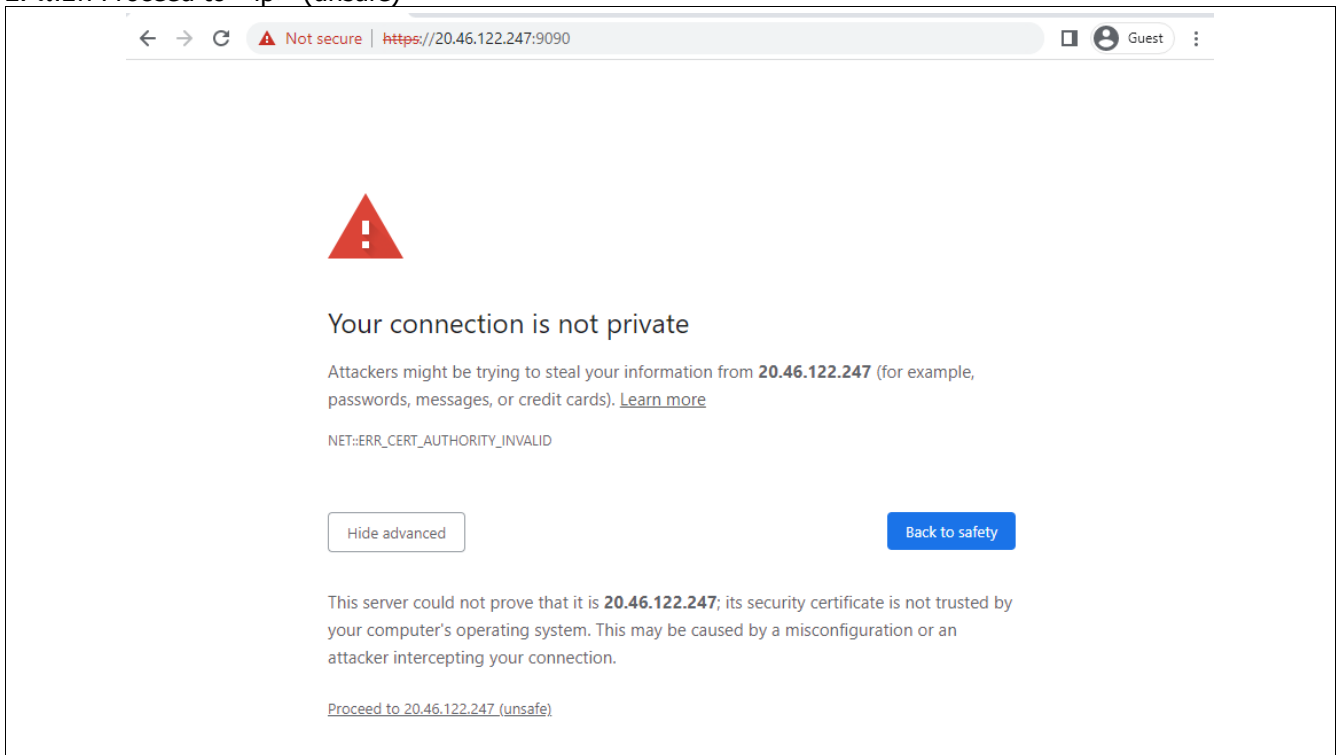
```
student@container-training-101:~$ sudo docker run -d --name myalpine1 \
    alpine@sha256:bc41182d7ef5ffc53a40b044e725193bc10142a1243f395ee852a8d9730fc2ad sleep 60
29acff33150d80cff67102d8d5e4fee456fed1ec3bfd84f527d6847af30184b3
student@container-training-101:~$ sudo docker container ls
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
29acff33150d alpine "sleep 60" 3 seconds ago Up 3 seconds myalpine1
student@container-training-101:~$
```

3. Dockerfile

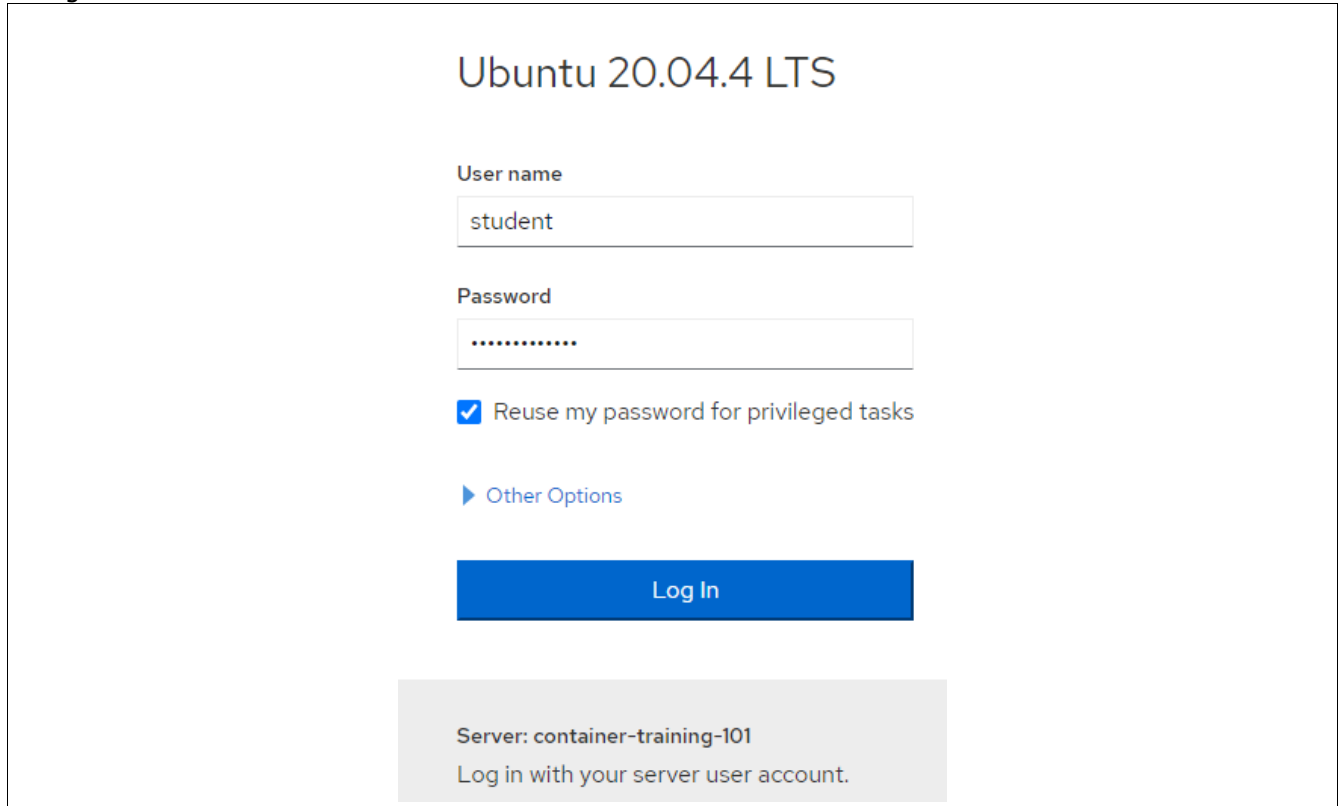
1. ทำการ login โดยเปิด Chrome browser และ browse ไปที่ <https://<ip>:9090> แทนที่ <ip> ด้วย ip ตามลำดับเครื่อง, เลือก Advanced



2. เลือก Proceed to <ip> (unsafe)



3. Log In ด้วย User name และ Password ที่กำหนด



Ubuntu 20.04.4 LTS

User name

student

Password

.....

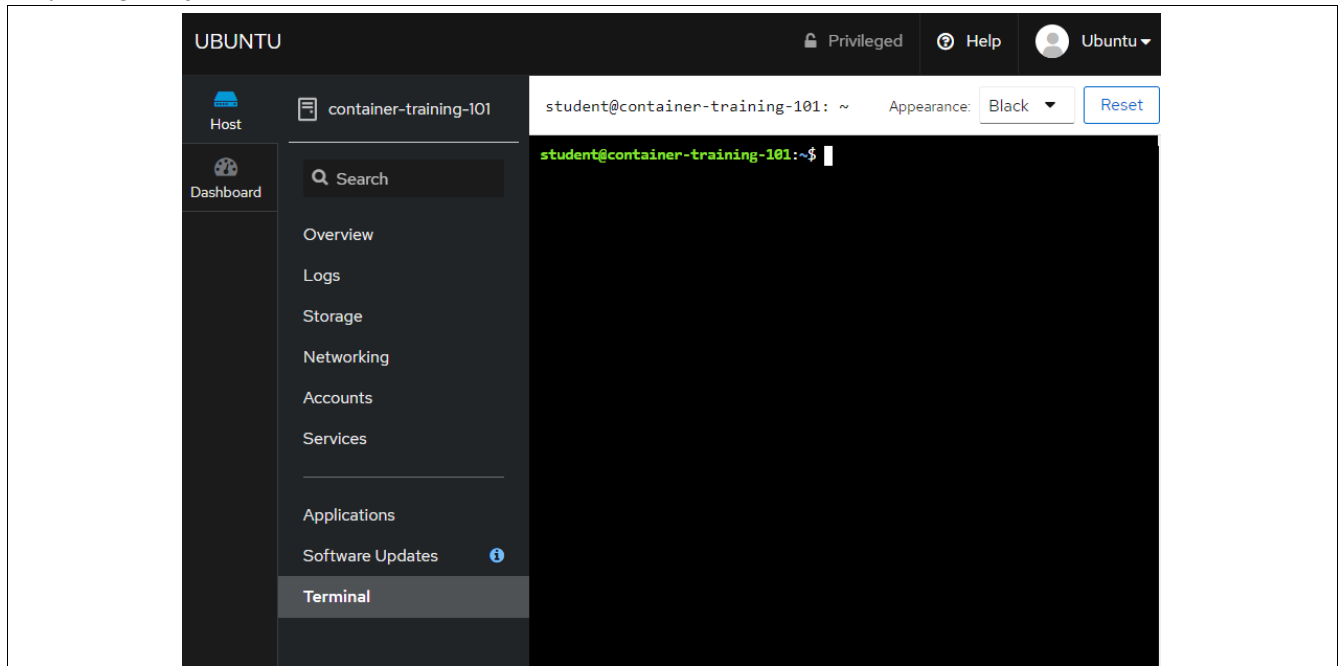
☒ Reuse my password for privileged tasks

[▶ Other Options](#)

Log In

Server: container-training-101
Log in with your server user account.

4. เลือก Terminal



5. สร้าง Folder ที่ Home ชื่อ myimage และสร้าง Dockerfile โดยใช้ Base image เป็น centos\ และให้รันคำสั่ง echo "Hello World"

```
student@container-training-101:~$ cd myimage
student@container-training-101:~/myimage$ sudo vi Dockerfile
FROM centos
CMD echo "Hello World"
```

6. ทำการ Build image จาก Dockerfile โดยใช้ tag ชื่อ myimage1

```
student@container-training-101:~/myimage$ sudo docker build -t myimage1 .
Sending build context to Docker daemon 2.048kB
Step 1/2 : FROM centos
---> 5d0da3dc9764
Step 2/2 : CMD echo "Hello World"
---> Running in fb6f9a217940
Removing intermediate container fb6f9a217940
---> b5448395bc84
Successfully built b5448395bc84
Successfully tagged myimage1:latest
student@container-training-101:~/myimage$ sudo docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
myimage1	latest	b5448395bc84	4 seconds ago	231MB
alpine	latest	9c6f07244728	3 weeks ago	5.54MB
hello-world	latest	feb5d9fea6a5	11 months ago	13.3kB
centos	latest	5d0da3dc9764	11 months ago	231MB
alpine	3.5	f80194ae2e0c	3 years ago	3.99MB

```
student@container-training-101:~/myimage$
```


7. สร้างไฟล์เพิ่มใน Build context สำหรับ image โดย Generate ไฟล์ขนาดใหญ่ชื่อ aa และ bb ตามลำดับ และให้อยู่ในโฟลเดอร์ myimage

```
student@container-training-101:~/myimage$ ls
Dockerfile

student@container-training-101:~/myimage$ sudo dd if=/dev/zero of=aa bs=1M count=50
50+0 records in
50+0 records out
52428800 bytes (52 MB, 50 MiB) copied, 0.0350696 s, 1.5 GB/s
student@container-training-101:~/myimage$ sudo dd if=/dev/zero of=bb bs=1M count=50
50+0 records in
50+0 records out
52428800 bytes (52 MB, 50 MiB) copied, 0.0345639 s, 1.5 GB/s
student@container-training-101:~/myimage$ ls -lah
total 101M
drwxr-xr-x 2 root  root  4.0K Aug 31 08:47 .
drwxr-xr-x 5 student student 4.0K Aug 31 08:23 ..
-rw-r--r-- 1 root  root   50M Aug 31 08:47 aa
-rw-r--r-- 1 root  root   50M Aug 31 08:47 bb
-rw-r--r-- 1 root  root   37 Aug 31 08:29 Dockerfile
student@container-training-101:~/myimage$
```

8. ทำการ Build image จาก Dockerfile โดยใช้ tag ชื่อ myimage2

```
student@container-training-101:~/myimage$ sudo docker build -t myimage2 .Sending build
context to Docker daemon 104.9MB
Step 1/2 : FROM centos
---> 5d0da3dc9764
Step 2/2 : CMD echo "Hello World"
---> Using cache
---> b5448395bc84
Successfully built b5448395bc84
Successfully tagged myimage2:latest
student@container-training-101:~/myimage$ sudo docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
myimage1	latest	b5448395bc84	17 minutes ago	231MB
myimage2	latest	b5448395bc84	17 minutes ago	231MB
alpine	latest	9c6f07244728	3 weeks ago	5.54MB
hello-world	latest	feb5d9fea6a5	11 months ago	13.3kB
centos	latest	5d0da3dc9764	11 months ago	231MB
alpine	3.5	f80194ae2e0c	3 years ago	3.99MB

```
student@container-training-101:~/myimage$
```

เปรียบเทียบขนาดของ Build context ของ myimage1 และ myimage2

9. สร้าง .dockerignore โดยใส่รายชื่อไฟล์ชื่อ aa เพื่อให้ถูก ignore จาก Build Context

```
student@container-training-101:~/myimage$ sudo vi .dockerignore
aa
```

10. ทำการ Build image จาก Dockerfile โดยใส่ tag ชื่อ myimage3 จะเห็นความเปลี่ยนแปลงของขนาดของ Build Context

```
student@container-training-101:~/myimage$ cat .dockerignore
aa
student@container-training-101:~/myimage$ sudo docker build -t myimage3 .
Sending build context to Docker daemon 52.43MB
Step 1/2 : FROM centos
--> 5d0da3dc9764
Step 2/2 : CMD echo "Hello World"
--> Using cache
--> b5448395bc84
Successfully built b5448395bc84
Successfully tagged myimage3:latest
student@container-training-101:~/myimage$ sudo docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
myimage1	latest	b5448395bc84	22 minutes ago	231MB
myimage2	latest	b5448395bc84	22 minutes ago	231MB
myimage3	latest	b5448395bc84	22 minutes ago	231MB
alpine	latest	9c6f07244728	3 weeks ago	5.54MB
hello-world	latest	feb5d9fea6a5	11 months ago	13.3kB
centos	latest	5d0da3dc9764	11 months ago	231MB
alpine	3.5	f80194ae2e0c	3 years ago	3.99MB

```
student@container-training-101:~/myimage$
```

11. รัน Container จาก myimage3 โดยเพิ่มคำสั่งแสดง ip ไปยัง docker run โดยตรง

```
student@container-training-101:~/myimage$ sudo docker run myimage3 ip address
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
45: eth0@if46: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
    link/ether 02:42:ac:11:00:03 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 172.17.0.3/16 brd 172.17.255.255 scope global eth0
        valid_lft forever preferred_lft forever
student@container-training-101:~/myimage$
```

12. สร้าง Dockerfile4 โดยบรรจุคำสั่งแสดง ip ไปใน Dockerfile จากนั้นทำการ Build เป็น image ชื่อ myimage4 และทดสอบรันอีกครั้ง

```
student@container-training-101:~/myimage$ sudo vi Dockerfile4
FROM centos
CMD ["ip", "address"]

student@container-training-101:~/myimage$ sudo docker build -f Dockerfile4 -t myimage4 .
Sending build context to Docker daemon 52.43MB
Step 1/2 : FROM centos
---> 5d0da3dc9764
Step 2/2 : CMD ["ip", "address"]
---> Running in 451207ae186c
Removing intermediate container 451207ae186c
---> 576dde9a8483
Successfully built 576dde9a8483
Successfully tagged myimage4:latest

student@container-training-101:~/myimage$ sudo docker run myimage4
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
47: eth0@if48: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
    link/ether 02:42:ac:11:00:03 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 172.17.0.3/16 brd 172.17.255.255 scope global eth0
        valid_lft forever preferred_lft forever
student@container-training-101:~/myimage$
```

13. สร้าง Dockerfile5 โดย EXPOSE port ที่ต้องการใช้งาน และทดสอบสถานะ

```

student@container-training-101:~/myimage$ sudo vi Dockerfile5
FROM centos
CMD ["sleep", "1000"]
EXPOSE 8000/tcp

student@container-training-101:~/myimage$ sudo docker build -f Dockerfile5 -t myimage5 .
Sending build context to Docker daemon 52.43MB
Step 1/3 : FROM centos
--> 5d0da3dc9764
Step 2/3 : CMD ["sleep", "1000"]
--> Running in 415c2fbfe159
Removing intermediate container 415c2fbfe159
--> 8e538a73de85
Step 3/3 : EXPOSE 8000/tcp
--> Running in b7e9bc962f67
Removing intermediate container b7e9bc962f67
--> 9006802807c2
Successfully built 9006802807c2
Successfully tagged myimage5:latest

student@container-training-101:~/myimage$ sudo docker run -d myimage5
fe8e2901293e7ee7a1545e11acf3e967e71a9dfafc633aed109653e568104960

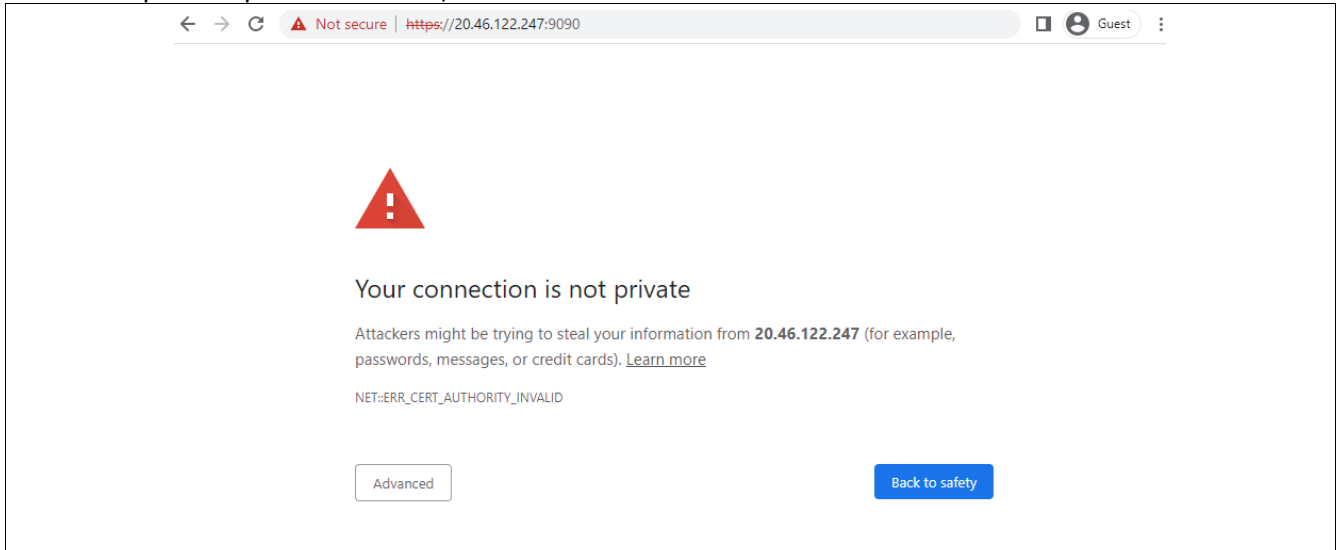
student@container-training-101:~/myimage$ sudo docker container ls

```

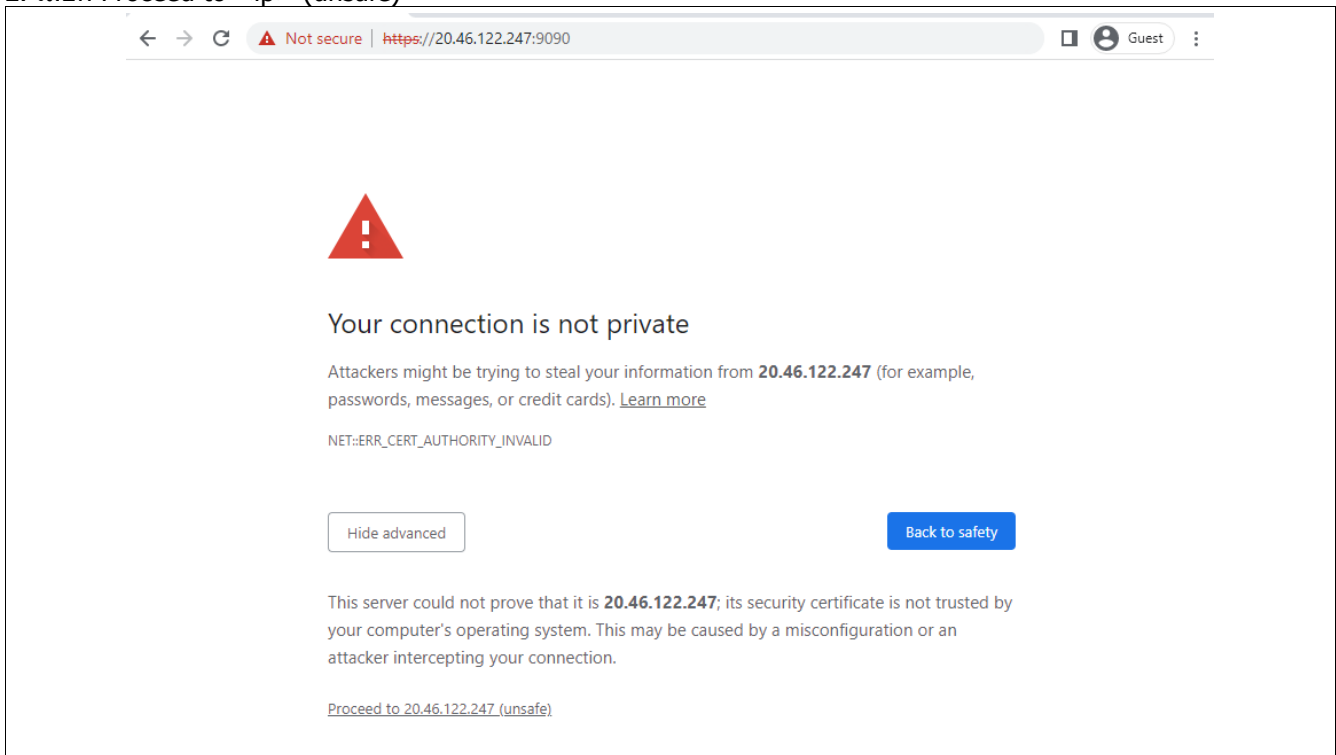
CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
fe8e2901293e	myimage5	"sleep 1000"	6 seconds ago	Up 6 seconds	8000/tcp	compassionate_hopper

4. Network

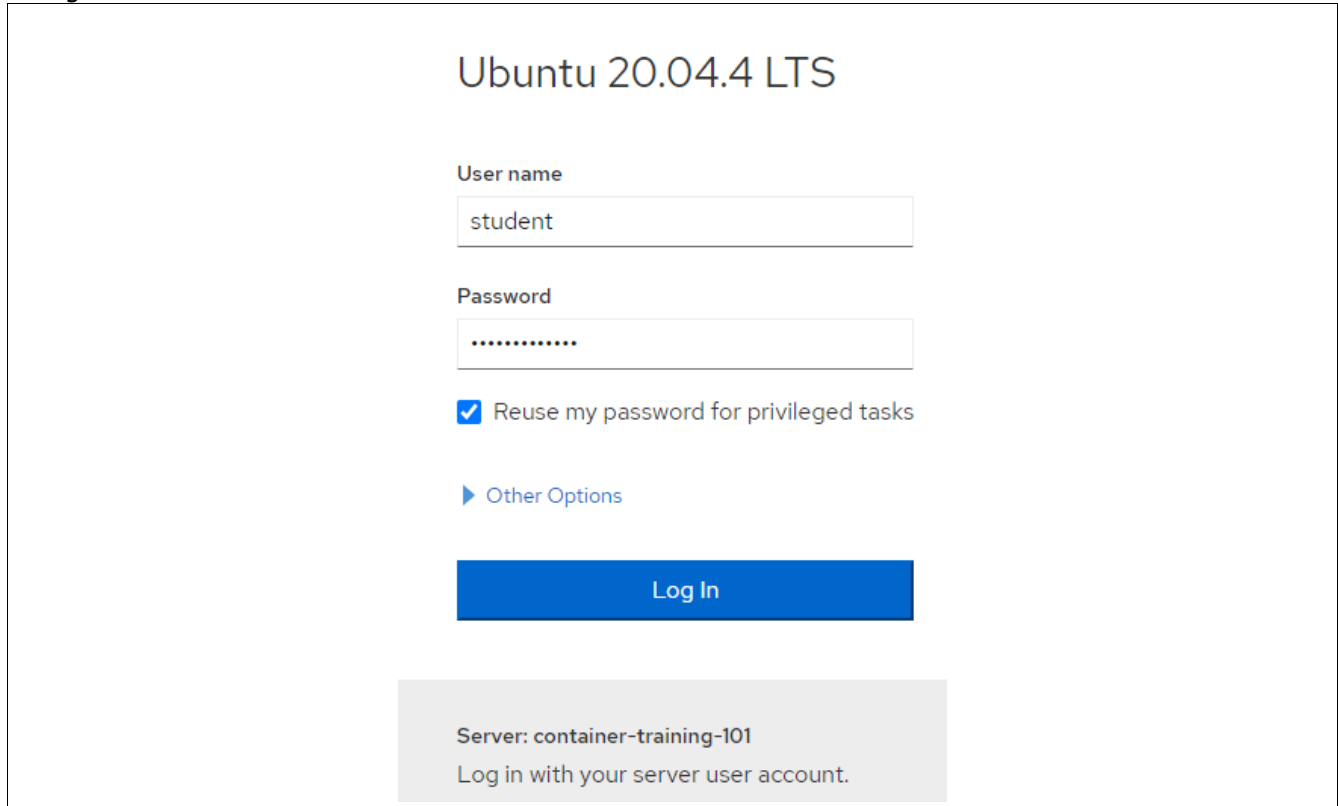
1. ทำการ login โดยเปิด Chrome browser และ browse ไปที่ <https://<ip>:9090> แทนที่ <ip> ด้วย ip ตามลำดับเครื่อง, เลือก Advanced



2. เลือก Proceed to <ip> (unsafe)



3. Log In ด้วย User name และ Password ที่กำหนด



Ubuntu 20.04.4 LTS

User name

student

Password

.....

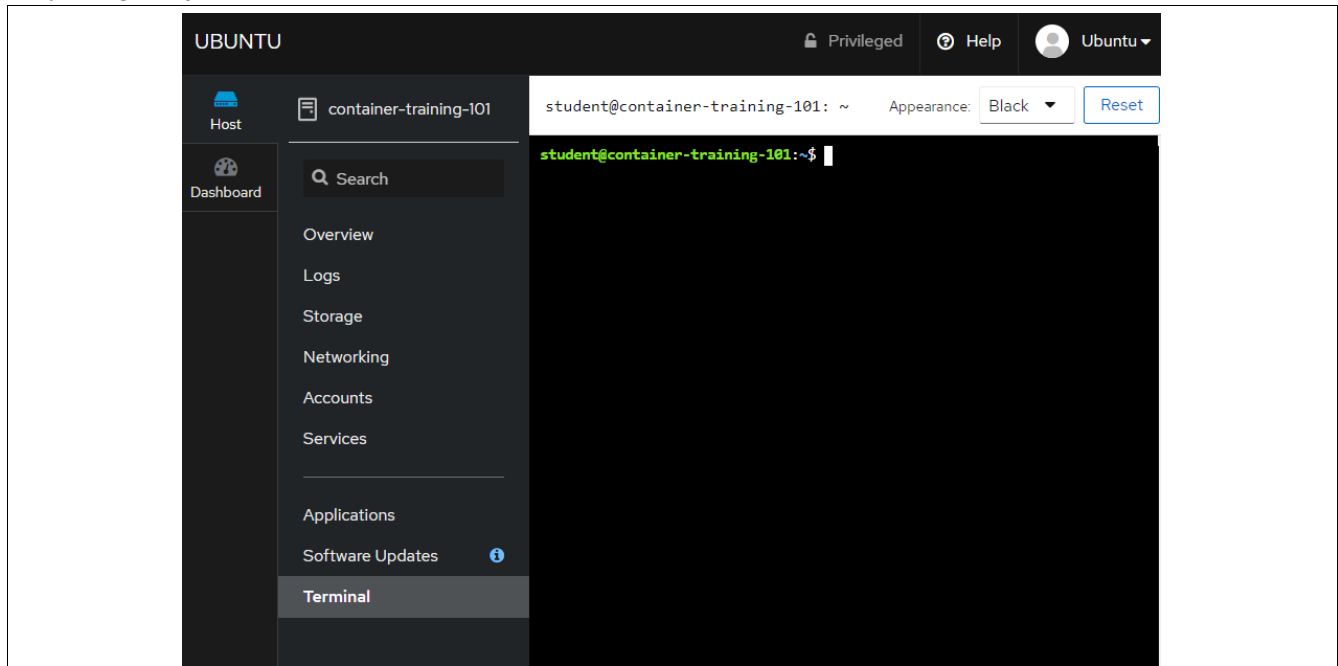
☒ Reuse my password for privileged tasks

[▶ Other Options](#)

Log In

Server: container-training-101
Log in with your server user account.

4. เลือก Terminal



5. สร้าง Container จำนวน 2 Containers โดยกำหนดให้ใช้ Network ร่วมกัน

เริ่มด้วยการสร้าง Container แรกโดยใช้ image nginx:alpine กำหนดให้ชื่อ myweb1 ทั้งนี้ nginx:alpine เป็น web server พร้อมใช้งานที่ port 80

```
student@container-training-101:~$ sudo docker run -d --name myweb1 nginx:alpine
Unable to find image 'nginx:alpine' locally
alpine: Pulling from library/nginx
213ec9aee27d: Already exists
2546ae67167b: Pull complete
23b845224e13: Pull complete
9bd5732789a3: Pull complete
328309e59ded: Pull complete
b231d02e5150: Pull complete
Digest: sha256:082f8c10bd47b6acc8ef15ae61ae45dd8fde0e9f389a8b5cb23c37408642bf5d
Status: Downloaded newer image for nginx:alpine
81098370aec03fd79b7e6c33edd93bdf5cf49ad8a91b4639ac357b04a5539efc
student@container-training-101:~$ sudo docker container ls | grep myweb1
81098370aec0 nginx:alpine "/docker-entrypoint...." About a minute ago Up About a minute 80/tcp myweb1
student@container-training-101:~$
```

6. สร้าง Container ถัดมาโดยใช้ image centos กำหนดให้ชื่อ mycentos-curl-1 ทั้งนี้ตัว centos เองไม่มี web server รันอยู่ แต่จะรันคำสั่ง curl เพื่อทดสอบว่าสามารถเข้าถึง web server ได้หรือไม่

```
student@container-training-101:~$ sudo docker run --rm --name mycentos-curl-1 centos curl -s localhost
student@container-training-101:~$
```

7. สร้าง Container เหมือนกับข้อ 6. แต่เพิ่มการกำหนดให้ใช้ network เดียวกันกับ myweb1 จากนั้นทดสอบว่าสามารถเข้าถึง web server ที่ myweb1 ได้หรือไม่

```
student@container-training-101:~$ sudo docker run --rm --name mycentos-curl-2 --network=container:myweb1 centos
curl -s localhost
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
html { color-scheme: light dark; }
body { width: 35em; margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif; }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
student@container-training-101:~$
```

8. นอกจากการแชร์ network แล้ว ยังสามารถแชร์ PID ได้ด้วย โดยเปลี่ยนจาก network เป็น pid และทดสอบโดยรันคำสั่ง ps aux บน container mycentos-curl-3 ทำให้สามารถมองเห็น process nginx ซึ่งรันอยู่บน myweb1 ได้

```
student@container-training-101:~$ sudo docker run --rm --name mycentos-curl-3 --pid=container:myweb1 centos ps aux
USER      PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.0  0.0  6312 4652 ?        Ss   11:01   0:00 nginx: master process nginx -g daemon off;
101       32  0.0  0.0  6768 1816 ?        S    11:01   0:00 nginx: worker process
101       33  0.0  0.0  6768 1816 ?        S    11:01   0:00 nginx: worker process
root       34  0.0  0.0 47588 3600 ?        Rs   11:14   0:00 ps aux
student@container-training-101:~$
```


9. รัน Container โดยใช้ network default และทดสอบด้วยคำสั่ง ip address

```
student@container-training-101:~$ sudo docker run --name mynet-default alpine ip address
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
57: eth0@if58: <BROADCAST,MULTICAST,UP,LOWER_UP,M-DOWN> mtu 1500 qdisc noqueue state UP
    link/ether 02:42:ac:11:00:04 brd ff:ff:ff:ff:ff:ff
    inet 172.17.0.4/16 brd 172.17.255.255 scope global eth0
        valid_lft forever preferred_lft forever
```

10. รัน Container โดยใช้ network host และทดสอบด้วยคำสั่ง ip address

```
student@container-training-101:~$ sudo docker run --name mynet-host --network host alpine ip address
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP qlen 1000
    link/ether 00:22:48:e7:2d:5c brd ff:ff:ff:ff:ff:ff
    inet 172.21.0.4/24 brd 172.21.0.255 scope global eth0
        valid_lft forever preferred_lft forever
    inet6 fe80::222:48ff:fee7:2d5c/64 scope link
        valid_lft forever preferred_lft forever
3: enP23394s1: <BROADCAST,MULTICAST,UP,LOWER_UP800> mtu 1500 qdisc mq master eth0 state UP qlen 1000
    link/ether 00:22:48:e7:2d:5c brd ff:ff:ff:ff:ff:ff
4: docker0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP
    link/ether 02:42:73:3c:ce:6f brd ff:ff:ff:ff:ff:ff
    inet 172.17.0.1/16 brd 172.17.255.255 scope global docker0
        valid_lft forever preferred_lft forever
    inet6 fe80::42:73ff:fe3c:ce6f/64 scope link
        valid_lft forever preferred_lft forever
40: veth4fe8c70@if39: <BROADCAST,MULTICAST,UP,LOWER_UP,M-DOWN> mtu 1500 qdisc noqueue master docker0 state UP
    link/ether 22:17:a3:c0:61:4c brd ff:ff:ff:ff:ff:ff
    inet6 fe80::2017:a3ff:fec0:614c/64 scope link
        valid_lft forever preferred_lft forever
52: veth67a10a0@if51: <BROADCAST,MULTICAST,UP,LOWER_UP,M-DOWN> mtu 1500 qdisc noqueue master docker0 state UP
    link/ether 02:d6:00:33:0c:a6 brd ff:ff:ff:ff:ff:ff
    inet6 fe80::d6:ff:fe33:ca6/64 scope link
        valid_lft forever preferred_lft forever
```

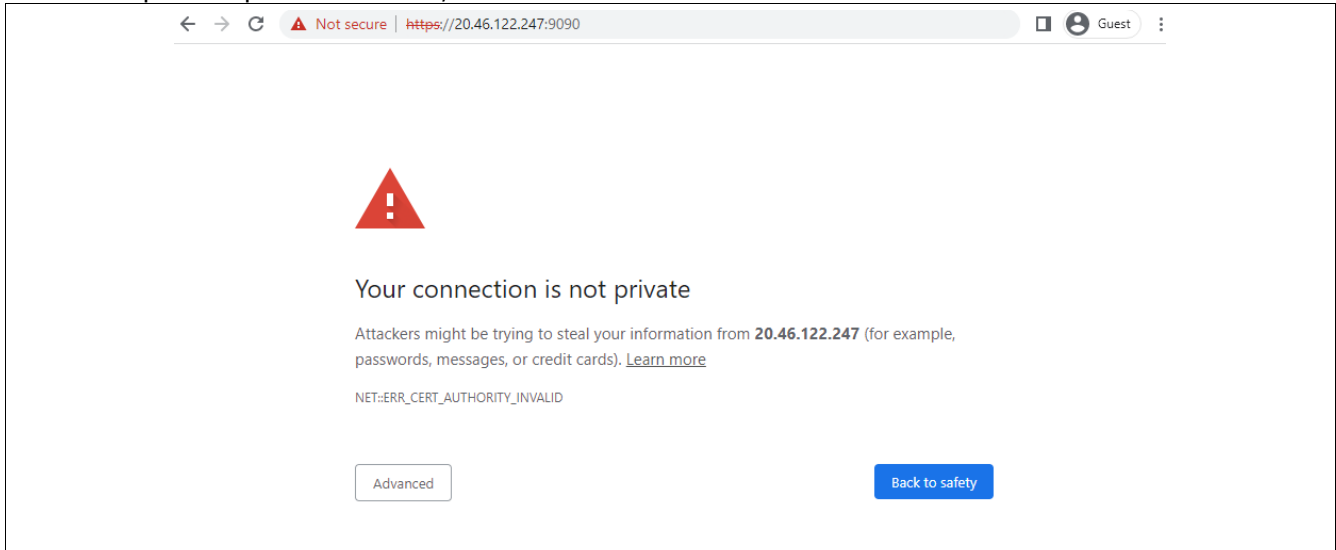
11. รันคำสั่ง ip address บน host

```
student@container-training-101:~$ ip address
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group default qlen 1000
    link/ether 00:22:48:e7:2d:5c brd ff:ff:ff:ff:ff:ff
    inet 172.21.0.4/24 brd 172.21.0.255 scope global eth0
        valid_lft forever preferred_lft forever
    inet6 fe80::222:48ff:fee7:2d5c/64 scope link
        valid_lft forever preferred_lft forever
3: enP23394s1: <BROADCAST,MULTICAST,SLAVE,UP,LOWER_UP> mtu 1500 qdisc mq master eth0 state UP group
default qlen 1000
    link/ether 00:22:48:e7:2d:5c brd ff:ff:ff:ff:ff:ff
    altname enP23394p0s2
4: docker0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
    link/ether 02:42:73:3c:ce:6f brd ff:ff:ff:ff:ff:ff
    inet 172.17.0.1/16 brd 172.17.255.255 scope global docker0
        valid_lft forever preferred_lft forever
    inet6 fe80::42:73ff:fe3c:ce6f/64 scope link
        valid_lft forever preferred_lft forever
40: veth4fe8c70@if39: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue master docker0 state
UP group default
    link/ether 22:17:a3:c0:61:4c brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet6 fe80::2017:a3ff:fec0:614c/64 scope link
        valid_lft forever preferred_lft forever
52: veth67a10a0@if51: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue master docker0 state
UP group default
    link/ether 02:d6:00:33:0c:a6 brd ff:ff:ff:ff:ff:ff link-netnsid 1
    inet6 fe80::d6:ff:fe33:ca6/64 scope link
        valid_lft forever preferred_lft forever
student@container-training-101:~$
```

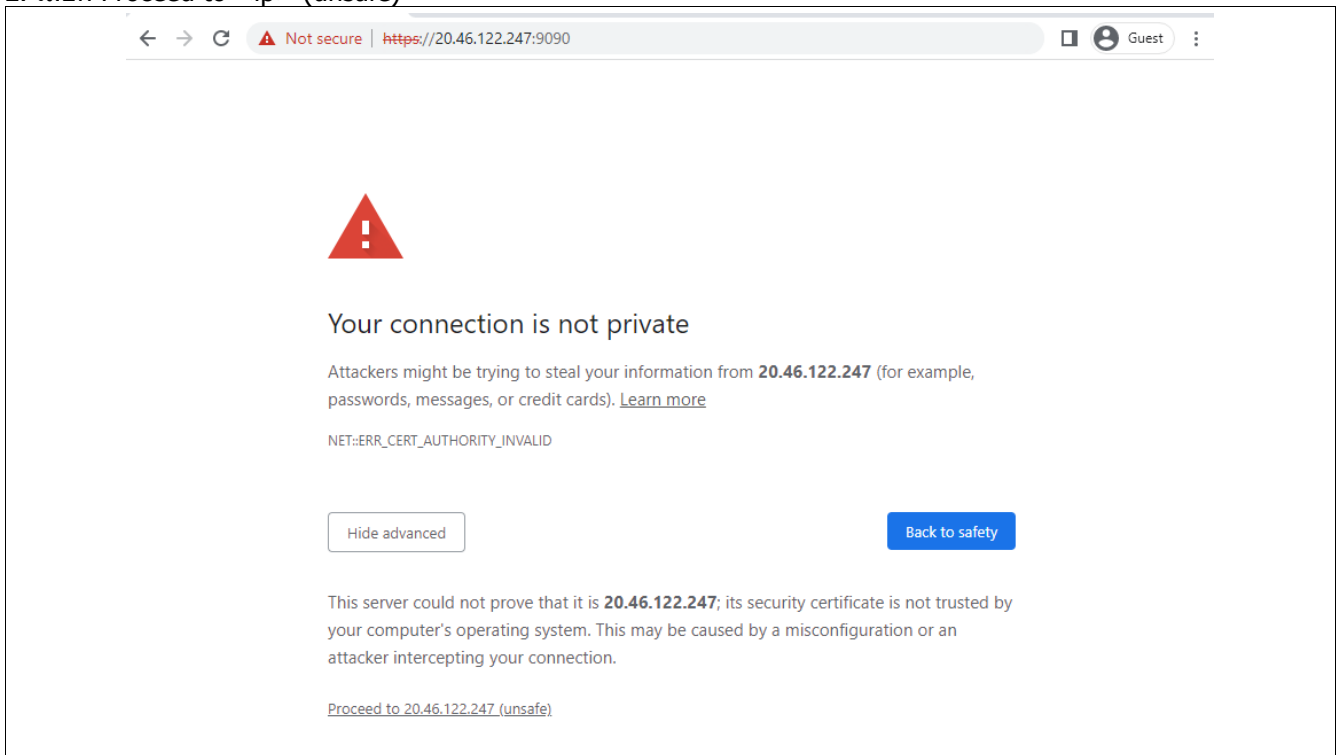
เปรียบเทียบผลลัพธ์ของคำสั่ง ip address จากข้อ 10-11

5. Bonus: Docker compose

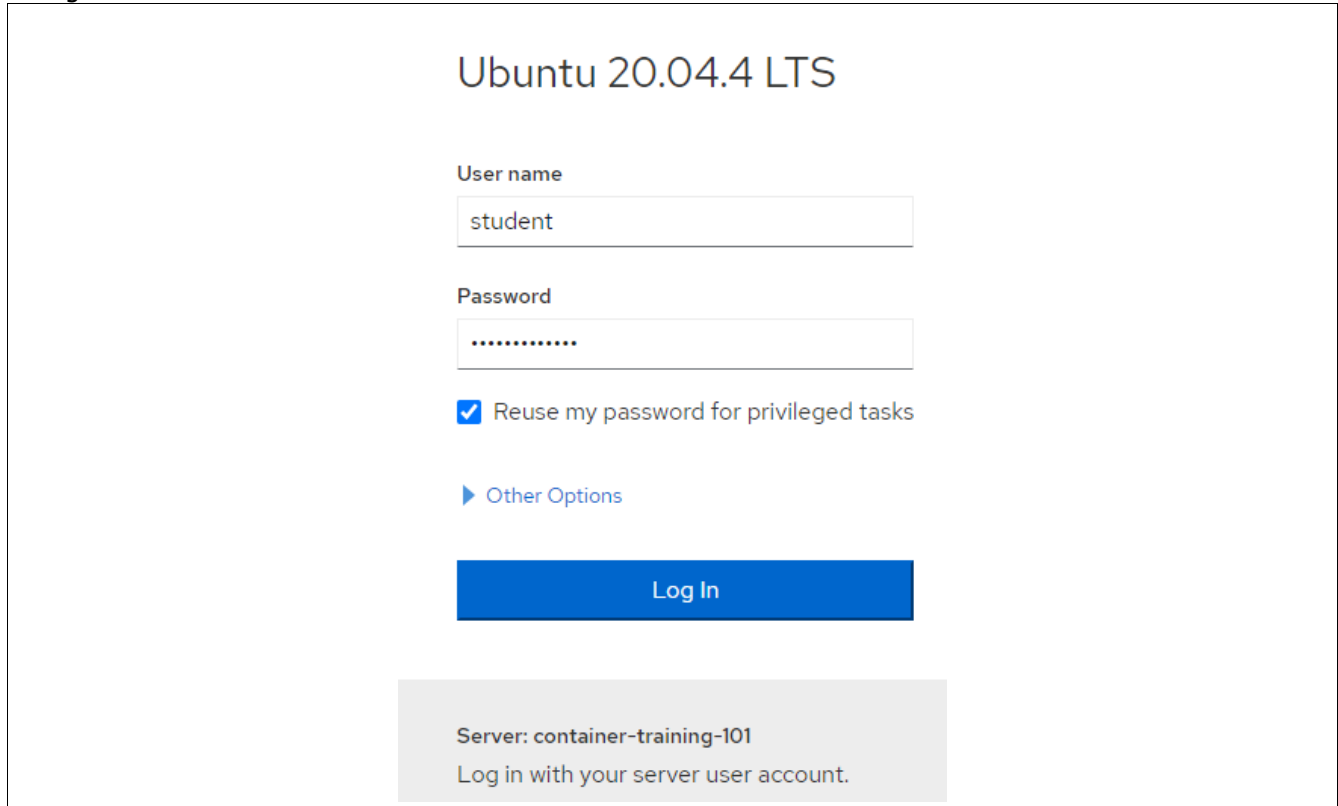
1. ทำการ login โดยเปิด Chrome browser และ browse ไปที่ <https://<ip>:9090>
แทนที่ <ip> ด้วย ip ตามลำดับเครื่อง, เลือก Advanced



2. เลือก Proceed to <ip> (unsafe)



3. Log In ด้วย User name และ Password ที่กำหนด



Ubuntu 20.04.4 LTS

User name

Password

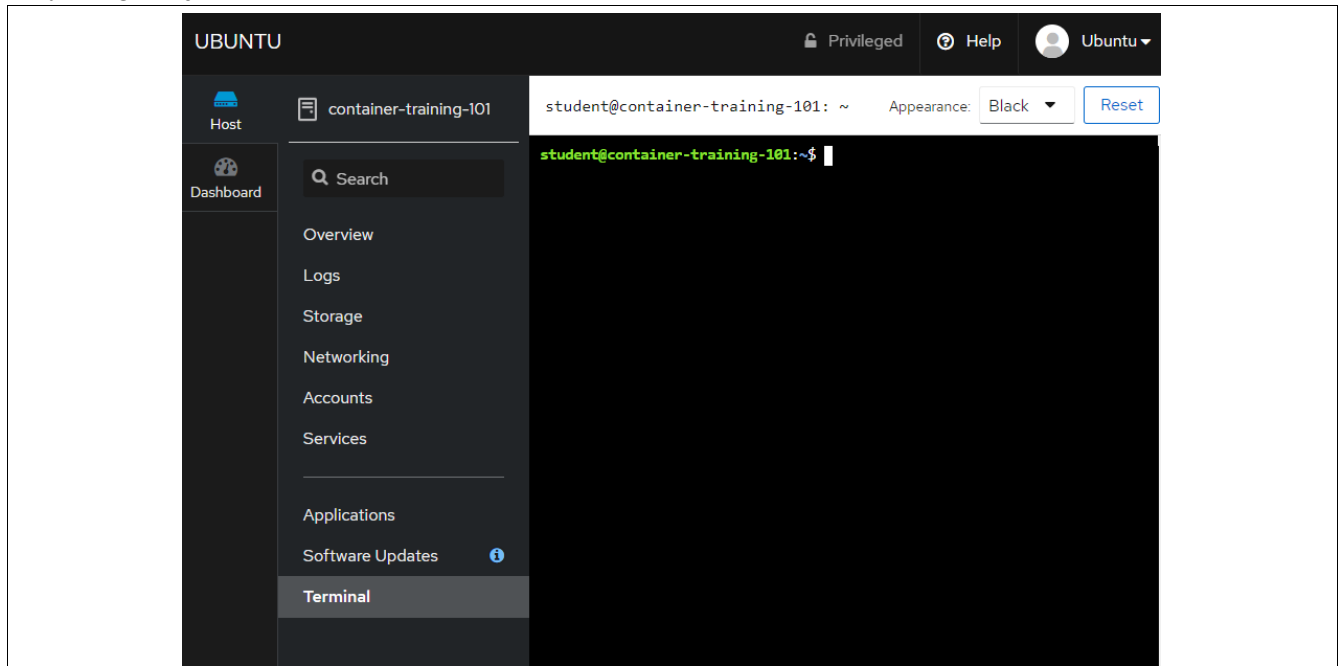
☒ Reuse my password for privileged tasks

[▶ Other Options](#)

[Log In](#)

Server: container-training-101
Log in with your server user account.

4. เลือก Terminal



5. สร้าง Folder my-compose และสร้าง Dockerfile สำหรับ httpd web services

```
student@container-training-101:~$ mkdir my-compose
student@container-training-101:~$ cd my-compose
student@container-training-101:~/my-compose$ sudo vi docker-compose.yaml
FROM httpd
```

6. สร้าง Folder my-compose และสร้างไฟล์ docker-compose.yaml

```
student@container-training-101:~/my-compose$ sudo vi docker-compose.yaml

version: "3.9" # optional since v1.27.0
services:
  web:
    build: .
    ports:
      - "8000:5000"
    volumes:
      - ./code
      - logvolume01:/var/log
    depends_on:
      - redis
  redis:
    image: redis
volumes:
  logvolume01: {}
```

7. รันด้วย docker compose up -d

```
student@container-training-101:~/my-compose$ sudo docker compose up -d
[+] Running 7/7
  :: redis Pulled                    5.5s
  :: 7a6db449b51b Already exists      0.0s
  :: 05b1f5f3b2c0 Pull complete      0.9s
  :: f0036f71a6fe Pull complete      1.1s
  :: cd7ddcecb993 Pull complete      1.7s
  :: 8cfc9a467ed7 Pull complete      1.8s
  :: 2a9998409df9 Pull complete      1.9s
[+] Running 3/3
  :: Network my-compose_default Cre... 0.1s
  :: Container my-compose-redis-1 S... 0.6s
  :: Container my-compose-web-1 Sta... 1.0s

student@container-training-101:~/my-compose$ sudo docker container ls
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                               NAMES
d07b132ab555   my-compose_web "httpd-foreground"      24 seconds ago Up 22 seconds 80/tcp,0.0.0.0:8000->5000/tcp, :::8000->5000/tcp my-compose-web-1
d4137af24c00   redis         "docker-entrypoint.s..." 24 seconds ago Up 22 seconds 6379/tcp my-compose-redis-1
```

8. ตรวจสอบ image ที่ถูกสร้างขึ้นจาก docker compose และ image ที่ docker compose ทำการ pull มาใช้งาน

```
student@container-training-101:~/my-compose$ sudo docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
myimage5      latest   9006802807c2  3 hours ago   231MB
myimage4      latest   576dde9a8483  3 hours ago   231MB
myimage1      latest   b5448395bc84  4 hours ago   231MB
myimage2      latest   b5448395bc84  4 hours ago   231MB
myimage3      latest   b5448395bc84  4 hours ago   231MB
redis         latest   dc7b40a0b05d  8 days ago    117MB
my-compose_web latest   7d741d686926  8 days ago    145MB
nginx         alpine   804f9cebfdc5  3 weeks ago   23.5MB
alpine        latest   9c6f07244728  3 weeks ago   5.54MB
hello-world   latest   feb5d9fea6a5  11 months ago 13.3kB
centos        latest   5d0da3dc9764  11 months ago 231MB
alpine        3.5     f80194ae2e0c  3 years ago   3.99MB
```

Image ที่ถูกสร้างขึ้นมาในขั้นตอนการทำ docker compose คือ image ไດ

End of Document