Lesson 06 Demo 01

Building a Secure Docker Container

Objective: To secure Docker containers by granting access to a non-root user within the container to mitigate the risks associated with running processes as the root user

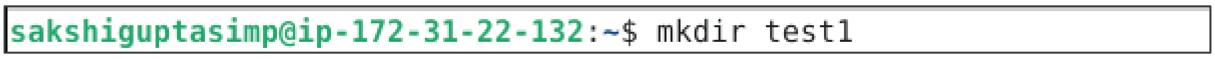
Tools required: Docker

Prerequisites: None

Steps to be followed:
1. Create a Dockerfile with a non-root user
2. Build the Docker image
3. Create and run a Docker container

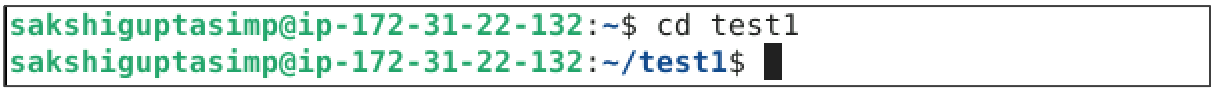
Step 1: Create a Dockerfile with a non-root user

1.1 Create a directory for the Dockerfile using the following command:
mkdir test1



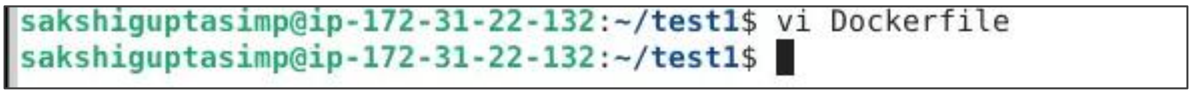
```
sakshiguptasimp@ip-172-31-22-132:~$ mkdir test1
```

1.2 Navigate inside the created directory using the following command:
cd test1



```
sakshiguptasimp@ip-172-31-22-132:~$ cd test1
sakshiguptasimp@ip-172-31-22-132:~/test1$
```

1.3 Create a Dockerfile using the following command:
vi Dockerfile



```
sakshiguptasimp@ip-172-31-22-132:~/test1$ vi Dockerfile
sakshiguptasimp@ip-172-31-22-132:~/test1$
```

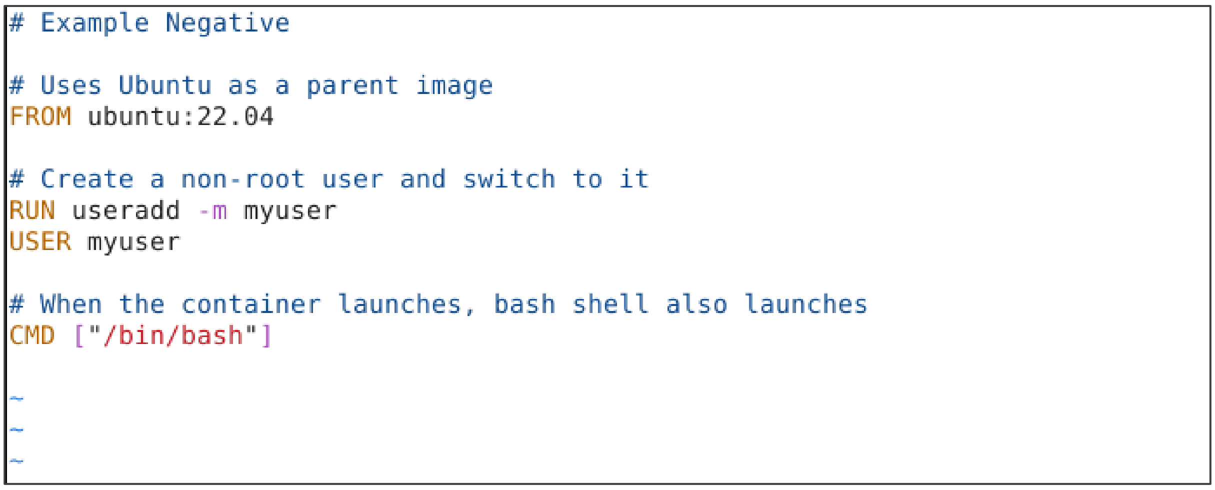
1.4 Edit the Dockerfile to create a non-root user named **myuser** using the following script:

# Use an official Ubuntu base image
FROM ubuntu:latest

# Create a new user called 'myuser'
RUN useradd -m myuser

# Set the user to 'myuser' for subsequent instructions
USER myuser

# The default command to run on the container start
CMD ["bash"]

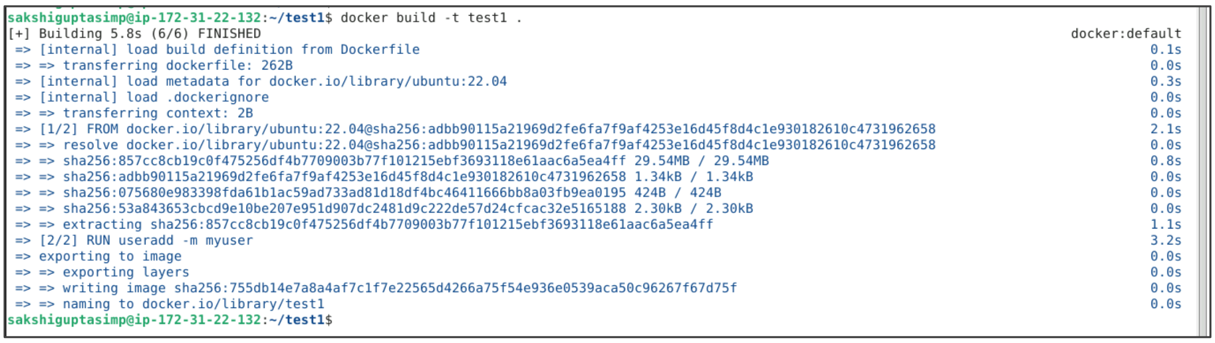


```
# Example Negative

# Uses Ubuntu as a parent image
FROM ubuntu:22.04

# Create a non-root user and switch to it
RUN useradd -m myuser
USER myuser

# When the container launches, bash shell also launches
CMD ["/bin/bash"]

~
~
~
```

This Dockerfile sets up a basic Ubuntu container, creates a new user named **myuser**, and switches to that user before performing any other operations.

Step 2: Build the Docker image

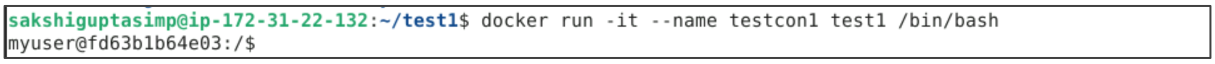
2.1 Build the Docker image using the following command:
docker build -t test1 .



```
sakshiguptasimp@ip-172-31-22-132:~/test1$ docker build -t test1 .
[+] Building 5.8s (6/6) FINISHED                                                                                                      docker:default
 => [internal] load build definition from Dockerfile                                                                                            0.1s
 => => transferring dockerfile: 262B                                                                                                            0.0s
 => [internal] load metadata for docker.io/library/ubuntu:22.04                                                                                 0.3s
 => [internal] load .dockerignore                                                                                                               0.0s
 => => transferring context: 2B                                                                                                                 0.0s
 => [1/2] FROM docker.io/library/ubuntu:22.04@sha256:adbb90115a21969d2fe6fa7f9af4253e16d45f8d4c1e930182610c4731962658                           2.1s
 => => resolve docker.io/library/ubuntu:22.04@sha256:adbb90115a21969d2fe6fa7f9af4253e16d45f8d4c1e930182610c4731962658                           0.0s
 => => sha256:857cc8cb19c0f475256df4b7709003b77f101215ebf3693118e61aac6a5ea4ff 29.54MB / 29.54MB                                                0.8s
 => => sha256:adbb90115a21969d2fe6fa7f9af4253e16d45f8d4c1e930182610c4731962658 1.34kB / 1.34kB                                                  0.0s
 => => sha256:075680e983398fda61b1ac59ad733ad81d18df4bc46411666bb8a03fb9ea0195 424B / 424B                                                      0.0s
 => => sha256:53a843653cbcd9e10be207e951d907dc2481d9c222de57d24cfcac32e5165188 2.30kB / 2.30kB                                                  0.0s
 => => extracting sha256:857cc8cb19c0f475256df4b7709003b77f101215ebf3693118e61aac6a5ea4ff                                                       1.1s
 => [2/2] RUN useradd -m myuser                                                                                                                 3.2s
 => exporting to image                                                                                                                          0.0s
 => => exporting layers                                                                                                                         0.0s
 => => writing image sha256:755db14e7a8a4af7c1f7e22565d4266a75f54e936e0539aca50c96267f67d75f                                                    0.0s
 => => naming to docker.io/library/test1                                                                                                        0.0s
sakshiguptasimp@ip-172-31-22-132:~/test1$
```

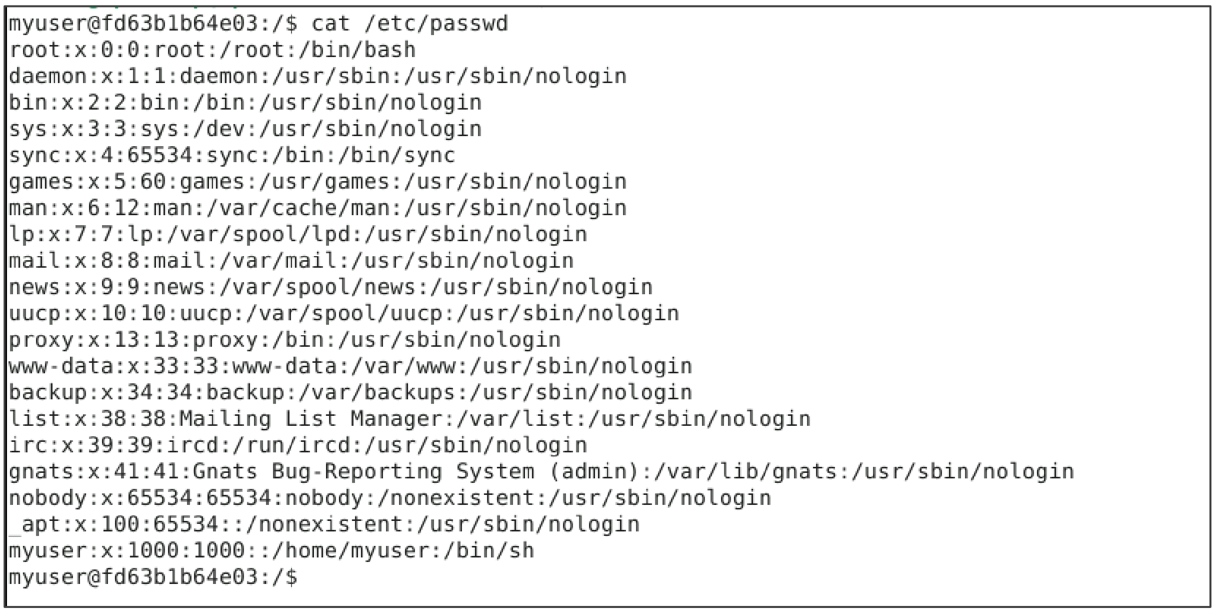
Step 3: Create and run a Docker container

3.1 From the newly built Docker image, create and run a Docker container using the following command:
docker run -it --name testcon1 test1



```
sakshiguptasimp@ip-172-31-22-132:~/test1$ docker run -it --name testcon1 test1 /bin/bash
myuser@fd63b1b64e03:/$
```

3.2 Verify that the container is running as **myuser** using the following command:
cat /etc/passwd



```
myuser@fd63b1b64e03:/$ cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
_apt:x:100:65534::/nonexistent:/usr/sbin/nologin
myuser:x:1000:1000::/home/myuser:/bin/sh
myuser@fd63b1b64e03:/$
```

This command displays the entry for **myuser** and confirms that the user exists within the container.

By following these steps, you have successfully secured Docker containers by granting access to a non-root user within the container, mitigating the risks associated with running processes as the root user.