CSIE 5310 Assignment 4 (Due on May 30th 14:10)

In this assignment, you will play with Docker container and QEMU. This is the last assignment of the course!

0. Late submission policy

- 1 pt deduction for late submissions within a day (before May 31st 14:10)
- 2 pts deduction for late submissions within 2 days (before Jun. 1st 14:10)
- zero points for submissions delayed by more than 2 days.

1. Part 1: Docker Container (60%)

In this section, you will learn how to dockerize any application by constructing a client-server model with containers using Docker. Please ensure that your work is compatible with **Docker Engine v23.0.0.**

You need to build two containers, one for the server and another for the client. The communication scenario between client/server is as follows:

1. Client makes a connection with the server
2. Server responds to the client with customized data, and save a copy of data into server volume
3. Client receives the data from the server and save the data into client volume

1-1. Client/Server Model Spec

The containers can be built starting from any Linux OS base images on Docker Hub. For client/server communication, feel free to adopt the programming language you are familiar with.

The container configuration of the client/server and the behavior of each container should comply with the below rules:

Note: `ID` stands for your student ID

- Server
  - Create a volume named `[ID]_servervol`
  - Mount the volume to `/[ID]_servervol`
  - Upon accepting the connection of the client, send a string as the format:
    - `"[ID] sending from the server"`, please send without double quotes
  - Save the above message as `sent.txt` in `/[ID]_servervol`
- Client
  - Create a volume named `[ID]_clientvol`
  - Mount the volume to `/[ID]_clientvol`
  - Make a connection with the server and receive the message, and save as `received.txt` in `/[ID]_clientvol`

1-2. Docker-compose

Docker-compose helps run multiple containers at once and in a specified context. You should leverage Docker-compose to deploy the two containers.
Try creating a `docker-compose.yml` to start the client/server model

1-3. Grading Method

We'll test your submission by running `docker compose up` in your root submission directory, and your `docker-compose.yml` should bring up two containers and make communication as the scenario mentioned above.

2. Part 2: QEMU (40%)

In this section of the assignment, your task is to modify **QEMU v7.0.0** (the version we use in previous assignments) so that it can bypass the Linux login authentication.

Recall that in previous assignments, we bypassed the login authentication by modifing the `/etc/passwd` file in the disk image:

> Next, open `/mnt/etc/passwd` , and update the first line to the following to disable root login password.
>
> ```
> root::0:0:root:/root:/bin/bash
> ```

However, for this assignment, you must boot the Linux kernel using the disk image without making any modifications to the `/etc/passwd` file.

To use your disk image built in previous assignments for this assignment, you can restore the first line in the `/etc/passwd` file to its original state:

```
root:x:0:0:root:/root:/bin/bash
```

or you can create a new disk image following the instructions from assignment 1 without modifying the `/etc/passwd` file. The kernel image should be compiled from the unmodified **Linux v5.15** source. You can use the `run-kvm.sh` script provided in assignment 1 to run the VM.

After completing these steps, you will find that you are unable to log in as the root user without entering the password. When you attempt to log in, you will see the following prompt:

```
Ubuntu 20.04.5 LTS ubuntu ttyAMA0

ubuntu login: root
Password:
```

Your goal is to modify QEMU to bypass this password authentication.

2-1. The Linux Login Program

The `login` program, located at `/bin/login` in the disk image, is responsible for handling user logins. It uses the `pam_authenticate()` function to verify user passwords. If the password is correct, the `pam_authenticate()` function returns zero, and the `login` program allows the user to log in.

To ensure that your `login` program is identical to that of the TA's, you should verify the md5 checksum of your `login` program, which should be the same as the TA's if you built the disk image by following the instructions in assignment 1. The md5 checksum of the TA's `login` program is:

```
# md5sum login
aeada2a4fdf5c3f62eb169c0617030d7  login
```

Your task is to modify QEMU so that it can intercept instruction(s) in the `login` program and manipulate the emulation in some way to bypass the authentication process.

You can use the following instruction to dump the disassembled code of the `login` program to a file and check where the `pam_authenticate()` function is invoked in the `login` program:

```
aarch64-linux-gnu-objdump -d login > FILE
```

This will allow you to analyze the code and identify the specific instruction(s) that you need to intercept and manipulate in order to bypass the authentication process.

2-2. Grading Method

You will receive full points if the TA is able to successfully log in to the VM as root using your QEMU implementation, either

1. by providing an arbitrary password or
2. without having to enter a password.

You will receive zero points if your patch fails to apply, your implementation fails to compile, or your implementation fails to meet the above requirements.

3. Homework submission

You should submit the assignment via NTU Cool.

Submission format

For part 1, pack your docker project into a directory named `[ID]_docker` , and the directory should be organized as:

```
[ID]_docker/
|
|--- docker-compose.yml
|
|--- server/
```

```
|    |
|    +-- Dockerfile
|    |
|    +-- server codes ...
|
+--- client/
     |
     +-- Dockerfile
     |
     +-- client codes ...
```

For part 2, prepare your patch named `[ID]_qemu.patch` of QEMU v7.0.0 that supports bypassing the login authentication. We will apply the patch to QEMU and test your code.

Finally, put `[ID]_docker` and `[ID]_qemu.patch` into a directory `[ID]_hw4`, and zip it into `[ID]_hw4.zip`, and submit it to NTU Cool.