CS-E4760: Platform Security (2023)
Exercise 1: Sandboxing

Part 1: Sandboxing in Theory

Answer the following questions:

1. **Process sandboxing.**

    a. Operating systems sandbox applications by providing each process with an independent view of memory. If this were not the case (that is, if all applications had identical page table mappings), then how could this be exploited by a malicious application? [2 points]

    b. Memory is commonly mapped using a 4KiB page size. Suggest an advantage and disadvantage of using a larger page size. [2 points]

    c. Pages can be mapped into the application's memory space with read, write, and execute permissions.

    What permissions would you give to pages containing the following, and why?
        i. Application code [2 points]
        ii. Shared library code [2 points]
        iii. Application data [2 points]

2. **VMs vs containers**

    a. A hypervisor provides an environment within which operating system kernels can run, whereas an operating system kernel provides an environment within which containers can run.

    Describe a situation where a virtual machine must be used instead of a container to sandbox two applications running on one machine. [2 points]

    b. Which component of the system interprets the application binary code in...
        i. A traditional virtual machine [1 point]
        ii. An application virtual machine [1 point]
        iii. A container system [1 point]

Part 2: Creating a virtual machine

If you already have a Linux system available with Docker installed, then feel free to skip this part.

First, install a **virtual machine manager**. You have several options here:

- VirtualBox (Windows, Linux, MacOS)
- virt-manager (Linux)

If you don't already have an opinion on this, then use VirtualBox.

Next, **download an installation image** of Ubuntu, and **create a new virtual machine**, connecting its optical drive to the image that you downloaded. Then, start the virtual machine, and complete Ubuntu's installation wizard.

Install Docker by following the instructions at https://docs.docker.com/engine/install/ubuntu/#install-using-the-repository, as well as your favourite text editor or development environment.

Part 3: Creating a Docker image

Next, you will write a basic Dockerfile containing a "Hello, World" application. Start by **writing an application** in your favourite programming language that prints the line "Hello, World!". The Bash shell will be installed with Ubuntu by default, but the choice of language is not important: use whatever you prefer.

You can then start your Dockerfile with

```
FROM ubuntu:latest
```

Next, **extend the Dockerfile** to do the following:

1. Insert your application source code into the container.
2. Install any software needed to build and run your application.
3. Build your application (if necessary).
4. Copy your application to an installation directory (e.g. /usr/local/bin or /app).
5. *When the container is run*, run the application.

Now, build your image with

```
$ docker build -t app .
```

and **run it** as follows:

```
$ docker run -it app
```

Submit the Dockerfile via MyCourses [15 points].

Submission

Submit to MyCourses your answers to the questions above as a text or PDF file, along with your Dockerfile from Part 3.