Docker security Hands-on labs for trainer

Hand out connection details where the playground is available with docker and docker compose installed if own environment is not an option.

ssh trainee1@ds-trainee1.westeurope.cloudapp.azure.com

NOTE:
The Kubernetes Fundamentals VM trainings image is used.

Exercise 1: Build the test docker image

Login to the terminal where docker is installed and get the sources from:

https://devon-training@dev.azure.com/devon-training/Training/_git/Docker_Security with git.

$ git clone https://devon-training@dev.azure.com/devon-training/Training/_git/Docker_Security *~/*
$ cd ~/Docker_Security/docker-security
$ docker compose up -d

Exercise 2: Get the docker build history

Get the docker build history and what do you notice?

All docker build steps are recorded.

Exercise 3: Get capabilities

Get into the docker container and get the capabilities of the current user and see what the capabilities are. You can search for all the capabilities meanings on https://man7.org/linux/man-pages/man7/capabilities.7.html

Investigate how you can get the capabilities if you don't know how to get this.

$ docker exec -ti test-container bash
$ capsh --uid=$(id -u) –print

Exercise 4: Get the capabilities of a process

Investigate how you can get the capabilities of the running processes if you don't know how to get this.

$ ps -auxf
$ getpcaps $pid

Exercise 5: Use ping inside the container

Try to run ping inside the container for example to localhost.
Why are you not allowed to do this?

$ ping localhost

*Because when you look inside the docker-compose.yml file, you will see that the capability **net_raw** is dropped, which is required to have for using ping.*

Exercise 6: Fix ping

Fix the ping issue and try to run ping again inside the container.

Remove the net_raw cap drop in the docker-compose.yml file and rebuild the image, exec into the new container and try again.

$ docker-compose down –volumes
$ docker rmi -f docker-security-docker-image-test:latest
$ docker compose up -d
$ docker exec -ti test-container bash
$ ping localhost

Exercise 7: Observe the /mount path

Go to the /mount path and observe the files that reside in there.

$ cd /mount
$ ls -alh

The .secret_pass file is also copied during the build. When coping full directories be careful not to also copy secrets inside the container.

Exercise 8: Print environment variables

Just for observability print out the environment variables and see what is in there.

$ printenv

Exercise 9: Login as root inside the container

It is very easy to gain root access inside the container by default, al you have to is login is the root user.

$ docker exec -ti -u 0 test-container bash

Exercise 10: Inspect the Dockerfile

What do you notice when you inspect the Dockerfile?

- *Old base image of Ubuntu*
- *Password set in an environment variable*
- *Installing not needed packages like: make, ssh, wget, curl, nano (These can be useful, but if it is not needed in the image, then do not install it. For this simple image it is not needed)*
- *Copying an entire folder can also copy files like secrets inside container, in this case the .secret_pass file is also copied inside the container*
- *CMD with tail is so that the container does not exit immediately after the container is created, because in order for the container to stay alive, there needs to be a running process.*

Bonus: AppArmor

Install bane, a AppArmor profile generator and use the sample AppArmor profile.
After running the container with the new profile activated try to run top and see what happens.
Also inspect the sample.toml (Later explained how to get the file) see what else you can and can't do.

Follow the docs for the installation instructions.

export
BANE_SHA256="69df3447cc79b028d4a435e151428bd85a816b3e26199cd010c74b7a17807a05"

*sudo curl -fSL "https://github.com/genuinetools/bane/releases/download/v0.4.4/bane-linux-amd64" -o "/usr/local/bin/bane" *
*&& echo "${BANE_SHA256} /usr/local/bin/bane" | sha256sum -c - *
&& sudo chmod a+x "/usr/local/bin/bane"

 # To get the sample AppArmor profile file run:
wget https://raw.githubusercontent.com/genuinetools/bane/master/sample.toml

#Get the status of AppArmor:

$ sudo aa-status

#Install the profile

sudo bane sample.toml

 # Information about using AppArmor

https://help.ubuntu.com/community/AppArmor

 # Alter docker-compose.yml file after container_name: xxx and add

```
security_opt:
  - apparmor:docker-nginx-sample
```

$ docker-compose down –volumes
$ docker rmi -f docker-security-docker-image-test:latest
$ docker compose up -d
$ docker exec -ti test-container bash
$ top