Création d'un petit environnement de cybersécurité avec Docker

Version de Debian : 12

1 – Installation de docker 1
2 – Création des fichiers Dockerfile 2
2.1 – Création du Dockerfile pour kali 2
2.2 – Création du Dockerfile pour metasploitable2 3
3 – Création du fichier docker-compose 3
4 – Création d'un petit script bash 5
5 – Vérification du bon fonctionnement 5

Le but de cette procédure va être d'expliquer la mise en œuvre d'un environnement propice à l'initiation au hacking à destination des débutants et à l'aide de Docker.

1 – Installation de docker

Afin d'installer Docker, il vous suffira de tapez les commandes suivantes dans l'ordre. Si vous souhaitez comprendre ce que font ces commandes n'hésitez pas à chercher sur Internet leur fonctionnement.

```
> sudo apt install ca-certificates curl gnupg lsb-release
> sudo mkdir -m 0755 -p /etc/apt/keyrings
> curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg
> echo \
  "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/debian \
  $(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
> sudo apt update
```

> sudo apt install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin docker-compose

2 – Création des fichiers Dockerfile

Les fichiers Dockerfile permettent de modifier des images existantes en les personnalisant selon nos souhaits. Le but ici va être de modifier l'image Kali de base car elle ne contient aucun tools et nous allons faire en sorte que celle-ci reste en écoute sur le port 22 afin de pouvoir s'y connecter en SSH.
De plus, nous allons créer un Dockerfile pour la machine metasploitable2 afin que celle-ci lance un script permettant le lancement des services vulnérables.

2.1 – Création du Dockerfile pour kali

Si vous ne souhaitez pas utiliser les commandes sudo à chaque commande, je vous conseil de passer en root. Tout d'abord, afin d'organiser les choses, nous allons créer un dossier dans lequel nous créerons notre Dockerfile.
> cd /root
> mkdir kalilinux
> cd kalilinux

Nous allons maintenant créer notre Dockerfile et rentrer les instructions présentes dans l'image ci-dessous
> mkdir Dockerfile

```
FROM kalilinux/kali-rolling # On pull une image déjà existant dans le registry Docker
RUN apt-get update -qq && apt install -qq -y openssh-server && apt install -qq -y kali-tools-top10 && apt clean # On va lancer différentes commandes
RUN mkdir /var/run/sshd # On va créer le dossier sshd pour le bon fonctonnement de SSH
RUN echo 'root:root123' | chpasswd # On va définir le mot de passe root
RUN sed -i 's/#PermitRootLogin prohibit-password/PermitRootLogin yes/' /etc/ssh/sshd_config # On va autoriser la connexion ssh en tant que root
EXPOSE 22/tcp # On va exposer le port 22 (port SSH)
CMD ["/usr/sbin/sshd", "-D"] # On va lancer cette commande en arrière plan au lancement du container
```

Maintenant, nous allons build notre image personnalisée. Pour ce faire, restez dans le même dossier que le Dockerfile et utilisez la commande suivante
> docker build -t kali-linux:v1.0 .

On peut maintenant lister nos images et vérifier que celle-ci est bien présente à l'aide de la commande suivante

```
root@DOCKER:~/docker/kali# docker image ls
REPOSITORY                TAG       IMAGE ID       CREATED        SIZE
metasploitable            v1.0      811b37f42421   9 days ago     1.51GB
kali-linux                v1.0      91f2cf7b4d03   9 days ago     4.08GB
kalilinux/kali-rolling    latest    4ffc8c9bd7b1   2 weeks ago    127MB
nginx                     latest    e4720093a3c1   3 weeks ago    187MB
debian                    latest    52f537fe0336   3 weeks ago    117MB
bkimminich/juice-shop     latest    402fefa2b068   8 weeks ago    621MB
portainer/portainer-ce    latest    1a0fb356ea35   3 months ago   294MB
tleemcjr/metasploitable2  latest    db90cb788ea1   6 years ago    1.51GB
```

2.2 – Création du Dockerfile pour metasploitable2

Les étapes vont être les mêmes que pour la création du Dockerfile de kali.

Commencez par créer un dossier
> cd /root
> mkdir metasploitable
> cd metasploitable

Nous allons maintenant créer notre Dockerfile et rentrer les instructions présentes dans l'image ci-dessous
> mkdir Dockerfile

```
FROM tleemcjr/metasploitable2
CMD sh -c /bin/services.sh && sleep infinity
```

Comme précédemment, nous allons build notre image
> docker build -t metasploitable:v1.0 .

On va vérifier que l'image est bien présente dans nos images existantes

```
root@DOCKER:~/docker/metasploitable# docker image ls
REPOSITORY                  TAG       IMAGE ID       CREATED         SIZE
metasploitable              v1.0      811b37f42421   9 days ago      1.51GB
kali-linux                  v1.0      91f2cf7b4d03   9 days ago      4.08GB
kalilinux/kali-rolling      latest    4ffc8c9bd7b1   2 weeks ago     127MB
nginx                       latest    e4720093a3c1   3 weeks ago     187MB
debian                      latest    52f537fe0336   3 weeks ago     117MB
bkimminich/juice-shop       latest    402fefa2b068   8 weeks ago     621MB
portainer/portainer-ce      latest    1a0fb356ea35   3 months ago    294MB
tleemcjr/metasploitable2    latest    db90cb788ea1   6 years ago     1.51GB
```

3 – Création du fichier docker-compose

Nous allons maintenant créer un fichier docker-compose.yaml afin d'orchestrer le lancement de tous ces containers.

Nous allons donc commencer par créer ce fichier
> cd /root
> nano docker-compose.yaml

Il vous suffira ensuite de rentrez les instructions comme l'image ci-dessous

```yaml
version: "3" #la version de docker-compose

services: #nos services, autrement dit tous les containers
  kali: # le container kali
    image: kali-linux:v1.0 # l'image à utiliser
    ports: #le mapping de port (doit être expose dans l'image)
      - "2222:22"
    container_name: kali # le nom qu'on souhaite donner au container
    restart: always # le container redémarrera si il s'arrête sans raison
    networks: # on spécifie le réseau à utiliser (on doit le déclarer plus bas)
      - pentest-lab
    hostname: kalilinux # le hostname de la machine

  metasploitable2:
    image: metasploitable:v1.0
    container_name: metasploitable2
    restart: always
    networks:
      - pentest-lab
    hostname: metasploitable2

  juie-shop:
    image: bkimminich/juice-shop
    ports:
      - "3000:3000"
    container_name: JuiceShop
    restart: always
    networks:
      pentest-lab:
        ipv4_address: 10.10.0.200 # on spécifie une ip fixe
    hostname: JuiceShop


networks: # on va ici déclarer le/les réseau(x)
  pentest-lab: # le nom de notre réseau
    driver: bridge # le type de réseau
    ipam:
      config: # permet de spécifier le/les configuration(s) du réseau
        - subnet: 10.10.0.0/24 # ici on spécifie le subnet de notre réseau
    driver_opts:
      com.docker.network.bridge.name: pentest-lab # permet de spécifier le nom du réseau lors d'un ifconfig ou ip a
```

Maintenant que ce fichier est créé, il vous faudra connaitre quelques commandes docker-compose afin d'administrer ces containers :

> docker-compose -f docker-compose.yaml up -d (permet d'exécuter les instructions de notre fichier docker-compose.yaml)

> docker-compose -f docker-compose.yaml down (supprime tous les containers existant dans le docker-compose.yaml)

> docker-compose -f docker-compose.yaml stop (arrête tous les containers existant dans le docker-compose.yaml)

> docker-composer -f docker-compose.yaml start (démarre tous les containers existant dans le docker-compose.yaml)

4 – Création d'un petit script bash

Ce petit script va avoir pour objectif de relancer les containers. Comme nous n'avons pas spécifier de volumes, les données seront supprimées et de nouveau container seront recréés.

```
#!/bin/bash

docker-compose -f /root/docker/docker-compose.yaml down
if [ $? -eq 0]; then
  docker-compose -f /root/docker/docker-compose.yaml up -d
else
  docker-compose -f /root/docker/docker-compose.yaml up -d
fi
```

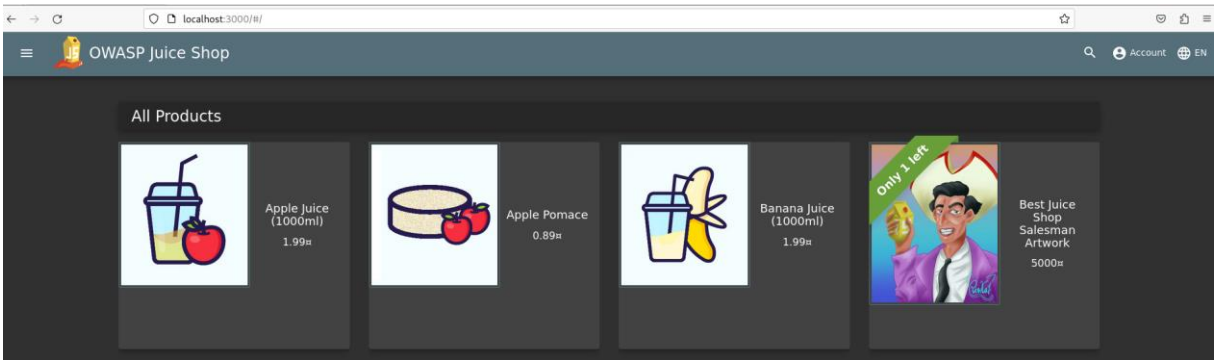
5 – Vérification du bon fonctionnement

Afin de vérifier que nos containers sont bien lancés, nous pouvons utiliser la commande suivante

> docker container ls

```
root@DOCKER:~/docker# docker container ls
CONTAINER ID   IMAGE                  COMMAND                  CREATED      STATUS          PORTS                                       NAMES
7247da6d2309   kali-linux:v1.0        "/usr/sbin/sshd -D"      8 days ago   Up 19 minutes   0.0.0.0:2222->22/tcp, :::2222->22/tcp       kali
0c6955bee990   bkimminich/juice-shop  "/nodejs/bin/node /j…"   8 days ago   Up 19 minutes   0.0.0.0:3000->3000/tcp, :::3000->3000/tcp   JuiceShop
0c97853696ea   metasploitable:v1.0    "/bin/sh -c 'sh -c /…"   8 days ago   Up 19 minutes                                               metasploitable2
```

On va donc vérifier que nous avons bien accès au site JuiceShop en se rendant sur notre localhost et sur le port spécifier dans le docker-compose (3000)



On va également vérifier que le container kali linux est bien accessible en SSH

```
root@DOCKER:~/docker# ssh root@127.0.0.1 -p 2222
The authenticity of host '[127.0.0.1]:2222 ([127.0.0.1]:2222)' can't be established.
ED25519 key fingerprint is SHA256:smeqFMdPGo6ht6jSXvBjK96aOxz4vvOEECiZSD/qpFA.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '[127.0.0.1]:2222' (ED25519) to the list of known hosts.
root@127.0.0.1's password:
Linux kalilinux 6.1.0-18-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.76-1 (2024-02-01) x86_64

The programs included with the Kali GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Kali GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
┌──(root㉿kalilinux)-[~]
└─#
```

Il ne reste plus qu'à vérifier que les services de la machine vulnérable sont bien en cours d'exécution, pour ce faire nous pouvons faire un scan nmap basique

```
┌──(root㉿kalilinux)-[~]
└─# nmap metasploitable2
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-03-11 12:30 UTC
Nmap scan report for metasploitable2 (10.10.0.3)
Host is up (0.0000050s latency).
rDNS record for 10.10.0.3: metasploitable2.docker_pentest-lab
Not shown: 981 closed tcp ports (reset)
PORT     STATE SERVICE
21/tcp   open  ftp
22/tcp   open  ssh
23/tcp   open  telnet
25/tcp   open  smtp
80/tcp   open  http
111/tcp  open  rpcbind
139/tcp  open  netbios-ssn
445/tcp  open  microsoft-ds
512/tcp  open  exec
513/tcp  open  login
514/tcp  open  shell
1099/tcp open  rmiregistry
1524/tcp open  ingreslock
2121/tcp open  ccproxy-ftp
3306/tcp open  mysql
5432/tcp open  postgresql
6667/tcp open  irc
8009/tcp open  ajp13
8180/tcp open  unknown
MAC Address: 02:42:0A:0A:00:03 (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 0.18 seconds
```

Tout est fonctionnel ! Si vous souhaitez réinitialiser l'environnement il vous suffira juste d'exécuter le script.

Fin de la procédure.