5 ‘chattr’ Commands to Make Important Files IMMUTABLE (Unchangeable) in Linux

BY NARAD SHRESTHA · OCTOBER 4, 2014

chattr (Change Attribute) is a command line Linux utility that is used to set/unset certain attributes to a file in Linux system to secure accidental deletion or modification of important files and folders, even though you are logged in as a root user.

In Linux native filesystems i.e. ext2, ext3, ext4, btrfs, etc. supports all the flags, though all the flags won’t support to all non-native FS. One cannot delete or modify file/folder once attributes are sets with chattr command, even though one have full permissions on it.

This is very useful to set attributes in system files like passwd and shadow files wherein user’s info are contains.

Syntax of chattr

```
# chattr [operator] [flags] [filename]
```

Attributes and Flags

Following are the list of common attributes and associated flags can be set/unset using the chattr command.

- If a file is accessed with ‘A‘ attribute set, its atime record is not updated.
- If a file is modified with ‘S‘ attribute set, the changes are updates synchronously on the disk.
- A file is set with ‘a‘ attribute, can only be open in append mode for writing.
- A file is set with ‘i‘ attribute, cannot be modified (immutable). Means no renaming, no symbolic link creation, no execution, no writable, only superuser can unset the attribute.
- A file with the ‘j‘ attribute is set, all of its information updated to the ext3 journal before being updated to the file itself.
- A file is set with ‘t‘ attribute, no tail-merging.
- A file with the attribute ‘d‘, will no more candidate for backup when the dump process is run.
- When a file has ‘u‘ attribute is deleted, its data are saved. This enables the user to ask for its undeletion.

Operator

- + : Adds the attribute to the existing attribute of the files.
- - : Removes the attribute to the existing attribute of the files.
- = : Keep the existing attributes that the files have.

Here, we are going to demonstrate some of the chattr command examples to set/unset attributes to a file and folders.

1. How to add attributes on files to secure from deletion

For demonstration purpose, we’ve used folder demo and file important_file.conf respectively. Before setting up attributes, make sure to verify that the existing files have any attributes set using ‘ls -l‘ command. Did you see the results, currently no attribute are set.

```
[root@tecmint tecmint]# ls -l
total 0
drwxr-xr-x. 2 root root 6 Aug 31 18:02 demo
-rwxrwxrwx. 1 root root 0 Aug 31 17:42 important_file.conf
```

To set attribute, we use the + sign and to unset use the - sign with the chattr command. So, let's set immutable bit on the files with +i flags to prevent anyone from deleting a file, even a root user don't have permission to delete it.

```
[root@tecmint tecmint]# chattr +i demo/
[root@tecmint tecmint]# chattr +i important_file.conf
```

Note: The immutable bit +i can only be set by superuser (i.e root) user or a user with sudo privileges can able to set.

After setting immutable bit, let's verify the attribute with command 'lsattr'.

```
[root@tecmint tecmint]# lsattr
----i----------- ./demo
----i----------- ./important_file.conf
```

Now, tried to delete forcefully, rename or change the permissions, but it won't allowed says "Operation not permitted".

```
[root@tecmint tecmint]# rm -rf demo/
rm: cannot remove âdemo/â: Operation not permitted
```

```
[root@tecmint tecmint]# mv demo/ demo_alter
mv: cannot move âdemo/â to âdemo_alterâ: Operation not permitted
```

```
[root@tecmint tecmint]# chmod 755 important_file.conf
chmod: changing permissions of âimportant_file.confâ: Operation not permitted
```

2. How to unset attribute on Files

In the above example, we've seen how to set attribute to secure and prevent files from a accidental deletion, here in this example, we will see how to reset (unset attribute) permissions and allows to make a files changeable or alterable using -i flag.

```
[root@tecmint tecmint]# chattr -i demo/ important_file.conf
```

After resetting permissions, verify the immutable status of files using 'lsattr' command.

```
[root@tecmint tecmint]# lsattr
---------------- ./demo
---------------- ./important_file.conf
```

You see in the above results that the '-i' flag removed, that means you can safely remove all the file and folder reside in tecmint folder.

```
[root@tecmint tecmint]# rm -rf *

[root@tecmint tecmint]# ls -l
total 0
```

3. How to Secure /etc/passwd and /etc/shadow files

Setting immutable attribute on files /etc/passwd or /etc/shadow, makes them secure from an accidental removal or tamper and also it will disable user account creation.

```
[root@tecmint tecmint]# chattr +i /etc/passwd
[root@tecmint tecmint]# chattr +i /etc/shadow
```

Now try to create a new system user, you will get error message saying ‘cannot open /etc/passwd‘.

```
[root@tecmint tecmint]# useradd tecmint
useradd: cannot open /etc/passwd
```

This way you can set immutable permissions on your important files or system configuration files to prevent from deletion.

4. Append data without Modifying existing data on a File

Suppose, you only want to allow everyone to just append data on a file without changing or modifying already entered data, you can use the ‘a‘ attribute as follows.

```
[root@tecmint tecmint]# chattr +a example.txt

[root@tecmint tecmint]# lsattr example.txt
-----a---------- example.txt
```

After setting append mode, the file can be opened for writing data in append mode only. You can unset the append attribute as follows.

```
[root@tecmint tecmint]# chattr -a example.txt
```

Now try to replace already existing content on a file example.txt, you will get error saying ‘Operation not permitted‘.

```
[root@tecmint tecmint]# echo "replace contain on file." > example.txt
-bash: example.txt: Operation not permitted
```

Now try to append new content on a existing file example.txt and verify it.

```
[root@tecmint tecmint]# echo "replace contain on file." >> example.txt
```

```
[root@tecmint tecmint]# cat example.txt
Here is the example to test 'a' attribute mean append only.
replace contain on file.
```

5. How to Secure Directories

To secure entire directory and its files, we use ‘-R‘ (recursively) switch with ‘+i‘ flag along with full path of the folder.

```
[root@tecmint tecmint]# chattr -R +i myfolder
```

After setting recursively attribute, try to delete the folder and its files.

```
[root@tecmint tecmint]# rm -rf myfolder/
rm: cannot remove 'myfolder/': Operation not permitted
```

To unset permission, we use same ‘-R’ (recursively) switch with ‘-i’ flag along with full path of the folder.

```
[root@tecmint tecmint]# chattr -R -i myfolder
```

That’s it! To know more about chattr command attributes, flags and options use the man pages.