
Chapter 9. working with files

In this chapter we learn how to recognise, create, remove, copy and move files using commands like **file**, **touch**, **rm**, **cp**, **mv** and **rename**.

9.1. all files are case sensitive

Files on Linux (or any Unix) are **case sensitive**. This means that **FILE1** is different from **file1**, and **/etc/hosts** is different from **/etc/Hosts** (the latter one does not exist on a typical Linux computer).

This screenshot shows the difference between two files, one with upper case **W**, the other with lower case **w**.

```
paul@laika:~/Linux$ ls
winter.txt  Winter.txt
paul@laika:~/Linux$ cat winter.txt
It is cold.
paul@laika:~/Linux$ cat Winter.txt
It is very cold!
```

9.2. everything is a file

A **directory** is a special kind of **file**, but it is still a (case sensitive!) **file**. Each terminal window (for example **/dev/pts/4**), any hard disk or partition (for example **/dev/sdb1**) and any process are all represented somewhere in the **file system** as a **file**. It will become clear throughout this course that everything on Linux is a **file**.

9.3. file

The **file** utility determines the file type. Linux does not use extensions to determine the file type. The command line does not care whether a file ends in **.txt** or **.pdf**. As a system administrator, you should use the **file** command to determine the file type. Here are some examples on a typical Linux system.

```
paul@laika:~$ file pic33.png
pic33.png: PNG image data, 3840 x 1200, 8-bit/color RGBA, non-interlaced
paul@laika:~$ file /etc/passwd
/etc/passwd: ASCII text
paul@laika:~$ file HelloWorld.c
HelloWorld.c: ASCII C program text
```

The **file** command uses a magic file that contains patterns to recognise file types. The magic file is located in **/usr/share/file/magic**. Type **man 5 magic** for more information.

It is interesting to point out **file -s** for special files like those in **/dev** and **/proc**.

```
root@debian6-# file /dev/sda
/dev/sda: block special
root@debian6-# file -s /dev/sda
/dev/sda: x86 boot sector; partition 1: ID=0x83, active, starthead...
root@debian6-# file /proc/cpuinfo
/proc/cpuinfo: empty
root@debian6-# file -s /proc/cpuinfo
/proc/cpuinfo: ASCII C++ program text
```

9.4. touch

9.4.1. create an empty file

One easy way to create an empty file is with **touch**. (We will see many other ways for creating files later in this book.)

This screenshot starts with an empty directory, creates two files with **touch** and the lists those files.

```
paul@debian7:~$ ls -l
total 0
paul@debian7:~$ touch file42
paul@debian7:~$ touch file33
paul@debian7:~$ ls -l
total 0
-rw-r--r-- 1 paul paul 0 Oct 15 08:57 file33
-rw-r--r-- 1 paul paul 0 Oct 15 08:56 file42
paul@debian7:~$
```

9.4.2. touch -t

The **touch** command can set some properties while creating empty files. Can you determine what is set by looking at the next screenshot? If not, check the manual for **touch**.

```
paul@debian7:~$ touch -t 200505050000 SinkoDeMayo
paul@debian7:~$ touch -t 130207111630 BigBattle.txt
paul@debian7:~$ ls -l
total 0
-rw-r--r-- 1 paul paul 0 Jul 11 1302 BigBattle.txt
-rw-r--r-- 1 paul paul 0 Oct 15 08:57 file33
-rw-r--r-- 1 paul paul 0 Oct 15 08:56 file42
-rw-r--r-- 1 paul paul 0 May 5 2005 SinkoDeMayo
paul@debian7:~$
```

9.5. rm

9.5.1. remove forever

When you no longer need a file, use **rm** to remove it. Unlike some graphical user interfaces, the command line in general does not have a **waste bin** or **trash can** to recover files. When you use **rm** to remove a file, the file is gone. Therefore, be careful when removing files!

```
paul@debian7:~$ ls
BigBattle.txt file33 file42 SinkoDeMayo
paul@debian7:~$ rm BigBattle.txt
paul@debian7:~$ ls
file33 file42 SinkoDeMayo
paul@debian7:~$
```

9.5.2. rm -i

To prevent yourself from accidentally removing a file, you can type **rm -i**.

```
paul@debian7:~$ ls
file33 file42 SinkoDeMayo
paul@debian7:~$ rm -i file33
rm: remove regular empty file `file33'? yes
paul@debian7:~$ rm -i SinkoDeMayo
rm: remove regular empty file `SinkoDeMayo'? n
paul@debian7:~$ ls
file42 SinkoDeMayo
paul@debian7:~$
```

9.5.3. rm -rf

By default, **rm -r** will not remove non-empty directories. However **rm** accepts several options that will allow you to remove any directory. The **rm -rf** statement is famous because it will erase anything (providing that you have the permissions to do so). When you are logged on as root, be very careful with **rm -rf** (the **f** means **force** and the **r** means **recursive**) since being root implies that permissions don't apply to you. You can literally erase your entire file system by accident.

```
paul@debian7:~$ mkdir test
paul@debian7:~$ rm test
rm: cannot remove `test': Is a directory
paul@debian7:~$ rm -rf test
paul@debian7:~$ ls test
ls: cannot access test: No such file or directory
paul@debian7:~$
```

9.6. cp

9.6.1. copy one file

To copy a file, use **cp** with a source and a target argument.

```
paul@debian7:~$ ls
file42 SinkoDeMayo
paul@debian7:~$ cp file42 file42.copy
paul@debian7:~$ ls
file42 file42.copy SinkoDeMayo
```

9.6.2. copy to another directory

If the target is a directory, then the source files are copied to that target directory.

```
paul@debian7:~$ mkdir dir42
paul@debian7:~$ cp SinkoDeMayo dir42
paul@debian7:~$ ls dir42/
SinkoDeMayo
```

9.6.3. cp -r

To copy complete directories, use **cp -r** (the **-r** option forces **recursive** copying of all files in all subdirectories).

```
paul@debian7:~$ ls
dir42 file42 file42.copy SinkoDeMayo
paul@debian7:~$ cp -r dir42/ dir33
paul@debian7:~$ ls
dir33 dir42 file42 file42.copy SinkoDeMayo
paul@debian7:~$ ls dir33/
SinkoDeMayo
```

9.6.4. copy multiple files to directory

You can also use **cp** to copy multiple files into a directory. In this case, the last argument (a.k.a. the target) must be a directory.

```
paul@debian7:~$ cp file42 file42.copy SinkoDeMayo dir42/
paul@debian7:~$ ls dir42/
file42 file42.copy SinkoDeMayo
```

9.6.5. cp -i

To prevent **cp** from overwriting existing files, use the **-i** (for interactive) option.

```
paul@debian7:~$ cp SinkoDeMayo file42
paul@debian7:~$ cp SinkoDeMayo file42
paul@debian7:~$ cp -i SinkoDeMayo file42
cp: overwrite `file42'? n
paul@debian7:~$
```

9.7. mv

9.7.1. rename files with mv

Use **mv** to rename a file or to move the file to another directory.

```
paul@debian7:~$ ls
dir33 dir42 file42 file42.copy SinkoDeMayo
paul@debian7:~$ mv file42 file33
paul@debian7:~$ ls
dir33 dir42 file33 file42.copy SinkoDeMayo
paul@debian7:~$
```

When you need to rename only one file then **mv** is the preferred command to use.

9.7.2. rename directories with mv

The same **mv** command can be used to rename directories.

```
paul@debian7:~$ ls -l
total 8
drwxr-xr-x 2 paul paul 4096 Oct 15 09:36 dir33
drwxr-xr-x 2 paul paul 4096 Oct 15 09:36 dir42
-rw-r--r-- 1 paul paul  0 Oct 15 09:38 file33
-rw-r--r-- 1 paul paul  0 Oct 15 09:16 file42.copy
-rw-r--r-- 1 paul paul  0 May  5 2005 SinkoDeMayo
paul@debian7:~$ mv dir33 backup
paul@debian7:~$ ls -l
total 8
drwxr-xr-x 2 paul paul 4096 Oct 15 09:36 backup
drwxr-xr-x 2 paul paul 4096 Oct 15 09:36 dir42
-rw-r--r-- 1 paul paul  0 Oct 15 09:38 file33
-rw-r--r-- 1 paul paul  0 Oct 15 09:16 file42.copy
-rw-r--r-- 1 paul paul  0 May  5 2005 SinkoDeMayo
paul@debian7:~$
```

9.7.3. mv -i

The **mv** also has a **-i** switch similar to **cp** and **rm**.

this screenshot shows that **mv -i** will ask permission to overwrite an existing file.

```
paul@debian7:~$ mv -i file33 SinkoDeMayo
mv: overwrite `SinkoDeMayo'? no
paul@debian7:~$
```

9.8. rename

9.8.1. about rename

The **rename** command is one of the rare occasions where the Linux Fundamentals book has to make a distinction between Linux distributions. Almost every command in the **Fundamentals** part of this book works on almost every Linux computer. But **rename** is different.

Try to use **mv** whenever you need to rename only a couple of files.

9.8.2. rename on Debian/Ubuntu

The **rename** command on Debian uses regular expressions (regular expression or short regex are explained in a later chapter) to rename many files at once.

Below a **rename** example that switches all occurrences of **txt** to **png** for all file names ending in **.txt**.

```
paul@debian7:~/test42$ ls
abc.txt file33.txt file42.txt
paul@debian7:~/test42$ rename 's/\.txt/\.png/' *.txt
paul@debian7:~/test42$ ls
abc.png file33.png file42.png
```

This second example switches all (first) occurrences of **file** into **document** for all file names ending in **.png**.

```
paul@debian7:~/test42$ ls
abc.png file33.png file42.png
paul@debian7:~/test42$ rename 's/file/document/' *.png
paul@debian7:~/test42$ ls
abc.png document33.png document42.png
paul@debian7:~/test42$
```

9.8.3. rename on CentOS/RHEL/Fedora

On Red Hat Enterprise Linux, the syntax of **rename** is a bit different. The first example below renames all ***.conf** files replacing any occurrence of **.conf** with **.backup**.

```
[paul@centos7 ~]$ touch one.conf two.conf three.conf
[paul@centos7 ~]$ rename .conf .backup *.conf
[paul@centos7 ~]$ ls
one.backup three.backup two.backup
[paul@centos7 ~]$
```

The second example renames all (*) files replacing one with ONE.

```
[paul@centos7 ~]$ ls
one.backup three.backup two.backup
[paul@centos7 ~]$ rename one ONE *
[paul@centos7 ~]$ ls
ONE.backup three.backup two.backup
[paul@centos7 ~]$
```

9.9. practice: working with files

1. List the files in the /bin directory
2. Display the type of file of /bin/cat, /etc/passwd and /usr/bin/passwd.
- 3a. Download wolf.jpg and LinuxFun.pdf from <http://linux-training.be> (wget <http://linux-training.be/files/studentfiles/wolf.jpg> and wget <http://linux-training.be/files/books/LinuxFun.pdf>)

```
wget http://linux-training.be/files/studentfiles/wolf.jpg  
wget http://linux-training.be/files/studentfiles/wolf.png  
wget http://linux-training.be/files/books/LinuxFun.pdf
```

- 3b. Display the type of file of wolf.jpg and LinuxFun.pdf
- 3c. Rename wolf.jpg to wolf.pdf (use mv).
- 3d. Display the type of file of wolf.pdf and LinuxFun.pdf.
4. Create a directory ~/touched and enter it.
5. Create the files today.txt and yesterday.txt in touched.
6. Change the date on yesterday.txt to match yesterday's date.
7. Copy yesterday.txt to copy.yesterday.txt
8. Rename copy.yesterday.txt to kim
9. Create a directory called ~/testbackup and copy all files from ~/touched into it.
10. Use one command to remove the directory ~/testbackup and all files into it.
11. Create a directory ~/etcbackup and copy all *.conf files from /etc into it. Did you include all subdirectories of /etc ?
12. Use rename to rename all *.conf files to *.backup . (if you have more than one distro available, try it on all!)