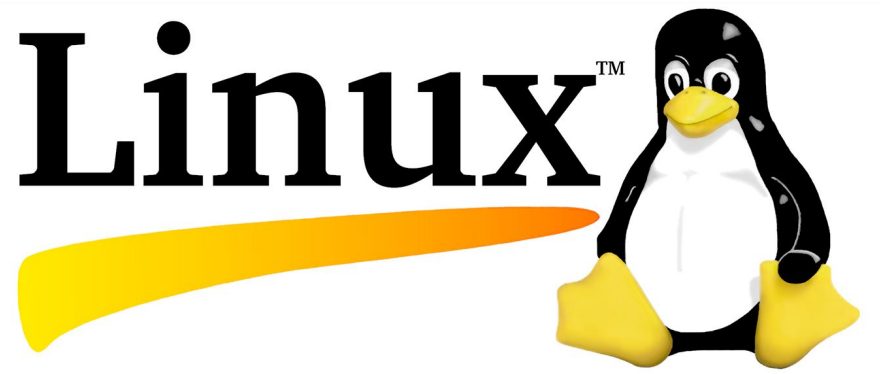
M. Mücahid Benlioğlu
Mert Koşan

Linux Tutorial



What is Linux?

Linux is a family of free and open-source software operating systems built around the Linux kernel, just like Windows and Mac OS.

Why Linux?

- *Free and open source:* Unlike Windows or Mac OS, source code for linux kernel is publicly available, along with many popular free distributions.
- *Flexibility:* Being open source also gives the advantage of customization to meet the specific needs of specific hardware, including desktops, servers, mobile phones and embedded systems.
- *Wide support:* Almost all programming languages and tools have support for Linux, some of which don't have proper support for other OSes. Moreover, there are many free (and mostly native to linux) libraries available to extend the operability. Also, built-in package managers introduce easy management of many of these libraries (*apt-get* on Ubuntu, *yum* on CentOS etc.)
- *Wide usage:* Because of aforementioned reasons, Linux is mostly preferred especially on servers, which is why every programmer should learn Linux :)

Commands

Here you can find short explanation and usage examples for some of the simple and commonly used commands. For a more compact cheat sheet, refer to this pdf. Originally retrieved from here.

***Tip*:** *Many commands have a "--help" switch. If you want to remember what a command does or what was its flags you can simply execute "<command> --help"*

Tip2: *There is another command called "man" which prints the manual page of other commands that contain detailed information. Execute "man <command>" to reach the manual page of a command.*

ls

List information about the files and folders (the current directory by default).

Sample Usage:

```
# list the contents of current directory
$ ls
folder  tuna

# list ALL contents (including hidden ones) of the current directory
$ ls -a
.  ..  .fbi_secrets  folder  tuna

# list all contents of the directory called "folder" in current directory
$ ls -a ./folder/
.  ..  file  .hidden_file

# list all contents of the current directory as list
$ ls -la
total 0
drwxrwxrwx 0 mbenlioglu mbenlioglu 512 Feb 21 02:54 .
drwxr-xr-x 0 mbenlioglu mbenlioglu 512 Feb 21 02:54 ..
-rw-rw-rw- 1 mbenlioglu mbenlioglu  23 Feb 21 02:47 .fbi_secrets
drwxrwxrwx 0 mbenlioglu mbenlioglu 512 Feb 20 17:28 folder
-rw-rw-rw- 1 mbenlioglu mbenlioglu  12 Feb 21 02:46 tuna
```

pwd

Prints working directory, that is the absolute path of the current directory

Sample Usage:

```
# prints the path of current directory
$ pwd
/home/mbenlioglu/f
```

whoami

Tells which user is currently active.

Sample Usage:

```
$ whoami
mertkosan
```

cd

Changes the current working directory to the desired directory.

Sample Usage:

```
$ pwd
/home/mbenlioglu/f
$ cd folder/
$ pwd
/home/mbenlioglu/f/folder
```

cat

Writes the contents of a file to standard output

Sample Usage:

```
$ ls -l
total 0
drwxrwxrwx 0 mbenlioglu mbenlioglu 512 Feb 21 03:00 folder
-rw-rw-rw- 1 mbenlioglu mbenlioglu  21 Feb 21 03:23 tuna
$ cat tuna
I like tuna sandwich
```

echo

Prints whatever is written right after it to standard output

Sample Usage:

```
$ echo Hello!
Hello!
$ echo "You're awesome"
You're awesome
```

touch

Creates an empty file.

```
$ touch newFile
$ ls
newFile
```

mkdir/rmdir

Creates/Removes a folder with given name.

Sample Usage:

```
$ ls -l
total 0
drwxrwxrwx 0 mbenlioglu mbenlioglu 512 Feb 21 03:00 folder
-rw-rw-rw- 1 mbenlioglu mbenlioglu  21 Feb 21 03:23 tuna
$ mkdir uselessFolder
$ ls -l
total 0
drwxrwxrwx 0 mbenlioglu mbenlioglu 512 Feb 21 03:00 folder
-rw-rw-rw- 1 mbenlioglu mbenlioglu  21 Feb 21 03:23 tuna
drwxrwxrwx 0 mbenlioglu mbenlioglu 512 Feb 21 03:30 uselessFolder
$ rmdir uselessFolder/
$ ls -l
total 0
drwxrwxrwx 0 mbenlioglu mbenlioglu 512 Feb 21 03:00 folder
-rw-rw-rw- 1 mbenlioglu mbenlioglu  21 Feb 21 03:23 tuna
```

cp/mv

Copies/Moves a file or directory to its new destination. (Tip: you can use mv to rename files)

Sample Usage:

```
# Copy "tuna" to ./folder
$ ls
cats  folder  tuna
$ ls ./folder/
file
$ cp tuna ./folder/
$ ls
cats  folder  tuna
$ ls ./folder/
file  tuna

# Move "tuna" from ./folder to current directory
$ ls
cats  folder
$ ls ./folder/
file  tuna
$ mv ./folder/tuna .
$ ls
cats  folder  tuna
$ ls ./folder/
file

# Rename "cats" to "dogs"
$ ls
cats  folder  tuna
$ mv cats dogs
$ ls
dogs  folder  tuna
```

rm

Removes a given file or folder.

Sample Usage:

```
$ ls
killMe  pieceOfTrash  someFile  uselessFolder
$ rm killMe
$ ls
pieceOfTrash  someFile  uselessFolder
$ ls ./uselessFolder/
uselessFile
$ rm uselessFolder/
rm: cannot remove 'uselessFolder/': Is a directory
$ rm -r uselessFolder/
$ ls
pieceOfTrash  someFile
$ ls ./pieceOfTrash/
readOnly-uselessFile
$ rm -r ./pieceOfTrash/
rm: remove write-protected regular file './pieceOfTrash/readOnly-uselessFile'?
rm: cannot remove './pieceOfTrash/': Directory not empty
$ rm -rf ./pieceOfTrash/
$ ls
someFile
```

head/tail

Prints some number of lines (default 10) from a large file to standard output.

Sample Usage:

```
$ ls -l
total 128
-rw-rw-rw- 1 mbenlioglu mbenlioglu    18 Feb 21 03:42 cats
drw-rw-rw- 0 mbenlioglu mbenlioglu   512 Feb 21 03:58 folder
-rw-rw-rw- 1 mbenlioglu mbenlioglu 92003 Feb 21 05:12 hugeFile
-rw-rw-rw- 1 mbenlioglu mbenlioglu    21 Feb 21 03:43 tuna

# Print first 10 lines of "hugeFile"
$ head hugeFile
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nunc vitae dolor
velit. Donec maximus, libero placerat convallis pretium, nibh metus cursus
ipsum, nec elementum elit tortor hendrerit turpis. Sed at sem pharetra, viverra
lorem non, condimentum elit. Donec at urna erat. Aenean vitae ante iaculis,
dignissim erat a, finibus nisi. Etiam orci ligula, egestas et eleifend sed,
vestibulum id metus. Curabitur vel nunc lectus. Duis fermentum egestas congue.
Nullam eu eros ut augue pretium varius. Cras eget aliquet ipsum. Curabitur
lacinia metus vitae erat tincidunt, et lacinia mauris porttitor. Mauris nisl
orci, interdum quis accumsan eu, euismod non mauris. Cras eget leo quis mauris
lacinia tincidunt. In volutpat velit id pretium condimentum. Aliquam congue

# Print first 3 lines of "hugeFile"
$ head hugeFile -n 3
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nunc vitae dolor
velit. Donec maximus, libero placerat convallis pretium, nibh metus cursus
ipsum, nec elementum elit tortor hendrerit turpis. Sed at sem pharetra, viverra

# Print last 3 lines of "hugeFile"
$ tail hugeFile -n 3
mollis justo turpis, ut rhoncus purus blandit quis. Suspendisse commodo dapibus
quam. Morbi lacinia augue quis tempor dignissim. Nulla tincidunt tempor est,
sit amet vestibulum nulla consectetur sit amet.
```

grep

Searches for a pattern in file or in standard input

Sample Usage:

```
$ ls
cats  contents  folder  hugeFile  tuna

# find and print the line containing "eros fermentum" in hugeFile
$ grep "eros fermentum" hugeFile
tortor in eros fermentum congue. Lorem ipsum dolor sit amet, consectetur
```

Some common operators

>

Writes the output of command to a file. Creates file if not exists, overwrites if exists.

Sample Usage:

```
$ ls -l > contents
$ cat contents
total 128
-rw-rw-rw- 1 mbenlioglu mbenlioglu    18 Feb 21 03:42 cats
-rw-rw-rw- 1 mbenlioglu mbenlioglu     0 Feb 21 05:24 contents
drw-rw-rw- 0 mbenlioglu mbenlioglu   512 Feb 21 03:58 folder
-rw-rw-rw- 1 mbenlioglu mbenlioglu 92003 Feb 21 05:12 hugeFile
-rw-rw-rw- 1 mbenlioglu mbenlioglu    21 Feb 21 03:43 tuna
```

>>

Writes the output of command to a file. Creates file if not exists, appends at the end of it if exists.

Sample Usage:

```
$ ls -l >> contents
$ cat contents
total 128
-rw-rw-rw- 1 mbenlioglu mbenlioglu    18 Feb 21 03:42 cats
-rw-rw-rw- 1 mbenlioglu mbenlioglu     0 Feb 21 05:24 contents
drw-rw-rw- 0 mbenlioglu mbenlioglu   512 Feb 21 03:58 folder
-rw-rw-rw- 1 mbenlioglu mbenlioglu 92003 Feb 21 05:12 hugeFile
-rw-rw-rw- 1 mbenlioglu mbenlioglu    21 Feb 21 03:43 tuna
$ echo "APPEND ME TO END" >> contents
$ cat contents
total 128
-rw-rw-rw- 1 mbenlioglu mbenlioglu    18 Feb 21 03:42 cats
-rw-rw-rw- 1 mbenlioglu mbenlioglu     0 Feb 21 05:24 contents
drw-rw-rw- 0 mbenlioglu mbenlioglu   512 Feb 21 03:58 folder
-rw-rw-rw- 1 mbenlioglu mbenlioglu 92003 Feb 21 05:12 hugeFile
-rw-rw-rw- 1 mbenlioglu mbenlioglu    21 Feb 21 03:43 tuna
APPEND ME TO END
```

&&

Executes the command on the right only if the command on the left executed successfully.

Sample Usage:

```
$ cat cats && echo "SUCESS!"
Cats are awesome!
SUCESS!
$ cat doesntExist && echo "SUCCESS!"
cat: doesntExist: No such file or directory
```

|

Pipe operator. Gives the output of left command as an input of right command.

Sample Usage:

```
# Give the output of "cat" command to "grep"
$ cat hugeFile | grep "eros fermentum"
tortor in eros fermentum congue. Lorem ipsum dolor sit amet, consectetur
```

Bonus:

You can make "hacker" looking commands by combining multiple commands with these operators :)

sudo

Makes operations as if the user is root. Some files are created by root or user cannot access it. Sudo will open the access to the users.

Sample Usage:

```
$ cat msgFromRoot
cat: msgFromRoot: Permission denied
$ sudo cat msgFromRoot
[sudo] password for mertkosan:
I am the Root!
```

chmod

Changes the permission of the files/folders. To examine chmod options and values of the permission, please refer to this website.

Sample Usage:

```
$ ls -l
total 0
-rw-rw-rw- 1 mertkosan mertkosan 0 Mar 16 16:04 newFile
$ chmod 777 newFile
$ ls -l
total 0
-rwxrwxrwx 1 mertkosan mertkosan 0 Mar 16 16:04 newFile
```

du

Shows size of the files and folders of a target directory.

Sample Usage:

```
$ ls
downloads  first  repos  second

# Show sizes of files and folders in the folder called "first" (in bytes)
$ du first/
214148  first/dataFolder/M6
225728  first/dataFolder/af_shell3
1024064 first/dataFolder/bone010
196032  first/dataFolder/coPapersDBLP
789952  first/dataFolder/hollywood-2009
4       first/dataFolder/karate
938496  first/dataFolder/nlpkkt120
8211648 first/dataFolder/nlpkkt240
987968  first/dataFolder/soc-LiveJournal1
3968    first/dataFolder/usroads
12592008        first/dataFolder
12592104        first/

# Show the humanized versions of total sizes (same as before but now sizes are in MB, GB etc.)
$ du -h first/
210M    first/dataFolder/M6
221M    first/dataFolder/af_shell3
1001M   first/dataFolder/bone010
192M    first/dataFolder/coPapersDBLP
772M    first/dataFolder/hollywood-2009
4.0K    first/dataFolder/karate
917M    first/dataFolder/nlpkkt120
7.9G    first/dataFolder/nlpkkt240
965M    first/dataFolder/soc-LiveJournal1
3.9M    first/dataFolder/usroads
13G     first/dataFolder
13G     first/

# Don't show each file & folder just show the total size of folder "first"
$ du -sch first/
13G     first/
13G     total
```

tar/gzip

Creates archives and compress the files/folders in order to make total size smaller which brings to easier mobility of files or folders.

Sample Usage:

```
# Compress folder named "data" using tar and gzip one after another
$ ls
data
# Size of the folder we will compress
$ du -sch data/
13G     data/
13G     total
$ tar -cf data.tar data/
$ ls
data.tar  data
$ gzip -9 data.tar
$ ls
data.tar.gz  data
# New size after compression
$ du -sch data.tar.gz
2.1G    data.tar.gz
2.1G    total

# Compress in single step
$ tar -czf data.tar.gz data/
$ ls
data.tar.gz  data

# Decompress and extract
$ ls
data.tar.gz
$ tar -xf data.tar.gz
$ ls
data.tar.gz  data
```

zip

Compress the files/folders with zip compression.

Sample Usage:

```
$ ls
newFile  newFolder
$ zip file.zip newFile newFolder/
  adding: newFile (stored 0%)
  adding: newFolder/ (stored 0%)
$ rm -r newFile newFolder/
$ ls
file.zip
$ unzip file.zip
Archive:  file.zip
 extracting: newFile
   creating: newFolder/
$ ls
file.zip  newFile  newFolder
```

wget

Retrieves files using HTTP, HTTPS, FTP and FTPS the most widely-used Internet protocols.

Sample Usage:

```
$ wget https://www.python.org/ftp/python/2.7/Python-2.7.tgz
--2018-03-16 16:31:02--  https://www.python.org/ftp/python/2.7/Python-2.7.tgz
Resolving www.python.org (www.python.org)... 151.101.36.223, 2a04:4e42:9::223
Connecting to www.python.org (www.python.org)|151.101.36.223|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 14026384 (13M) [application/octet-stream]
Saving to: ‘Python-2.7.tgz’

Python-2.7.tgz     100%[===============>]  13.38M  1.23MB/s    in 10s

2018-03-16 16:31:13 (1.31 MB/s) - ‘Python-2.7.tgz’ saved [14026384/14026384]

$ ls
Python-2.7.tgz
```

curl

Transfers data from Internet to terminal.

Sample Usage:

```
$ curl www.sabanciuniv.edu > sabanci.html
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--   100 58284    0 58284    0     0
$ ls
sabanci.html
```

ssh

Connect to a remote server and execute commands in it by using ssh connection protocol.

Sample Usage:

```
# Connect to flow.sabanciuniv.edu using ssh
mbenlioglu@BruceWayne:~$ ls
downloads  first  repos  second
# Print which OS we are currently using in detail
mbenlioglu@BruceWayne:~$ uname -a
Linux BruceWayne 4.4.0-43-Microsoft #1-Microsoft Wed Dec 31 14:42:53 PST 2014 x86_64 x86_64 x86_64 GNU/Linux
mbenlioglu@BruceWayne:~$ ssh mbenlioglu@flow.sabanciuniv.edu
mbenlioglu@flow.sabanciuniv.edu's password:
Last login: Thu Mar 15 13:28:33 2018 from 159.20.88.13
[mbenlioglu@flow ~]$ ls
bin  downloads  include  lib  repos  share  SUNetPassword.exe  var  WINDOWS
# Print which OS we are currently using in detail (notice it is different from our local machine)
[mbenlioglu@flow ~]$ uname -a
Linux flow.sabanciuniv.edu 2.6.18-419.el5 #1 SMP Fri Feb 24 22:47:42 UTC 2017 x86_64 x86_64 x86_64 GNU/Linux
```

sftp

Transfer files from a remote server to your local PC, or vica versa.

Sample Usage:

```
$ ls
doit.exe
# Open sftp connection then upload file to server
mbenlioglu@BruceWayne:~$ sftp mbenlioglu@flow.sabanciuniv.edu
mbenlioglu@flow.sabanciuniv.edu's password:
Connected to flow.sabanciuniv.edu.
sftp> put doit.exe .
Uploading doit.exe to /home/mbenlioglu/./doit.exe
doit.exe                                                                    100% 8600     8.4KB/s   00:01
sftp> ls
SUNetPassword.exe    WINDOWS           bin               doit.exe          downloads         include
lib                  repos             share             var
sftp> exit

# Open sftp conncetion then download a folder
mbenlioglu@BruceWayne:~$ sftp mbenlioglu@flow.sabanciuniv.edu
mbenlioglu@flow.sabanciuniv.edu's password:
Connected to flow.sabanciuniv.edu.
sftp> get -r downloads/ ./  #-r for recursive, which means get all files&folders inside "download/"
Fetching /home/mbenlioglu/downloads/ to ./downloads
Retrieving /home/mbenlioglu/downloads
sftp> exit
mbenlioglu@BruceWayne:~$ ls
doit.exe  downloads
```