Linux Fundamentals Part 2

Learned:

- The Secure Shell Protocol allows connection between devices which allows you to run commands on a machine.

- Hidden files can be found using flags that can reveal possible malware or suspicious files such as `-a` for showing all files in the directory.

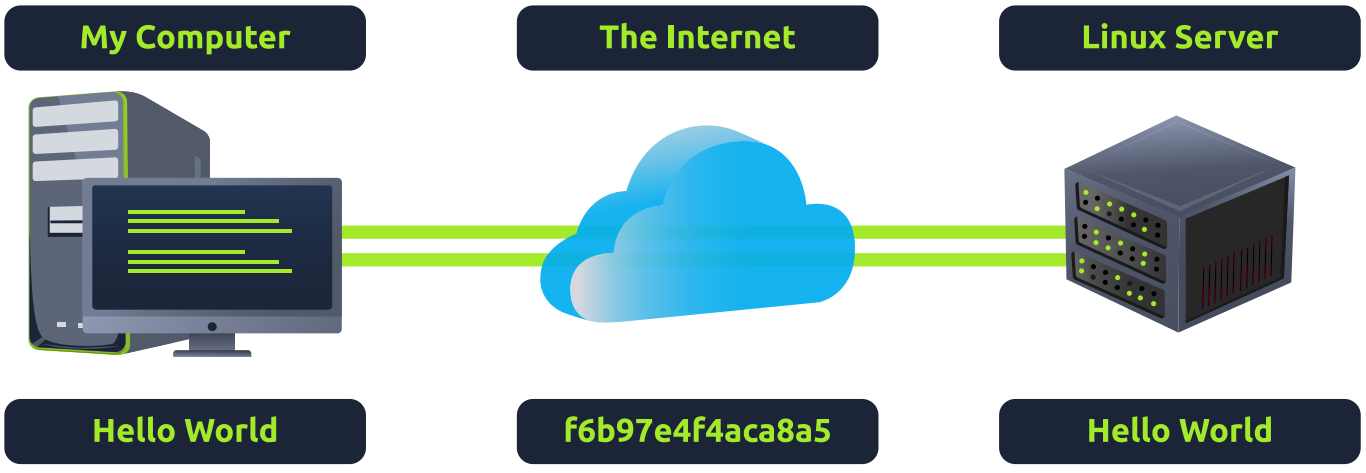
- Not all files can be accessed without having permissions and you can view file permissions by using `ls -h` .

- /etc, /var, /root, and /tmp directories show how and where data is stored on the device. They can include system configs, password-related files, logs, or temporary data.

Linux Fundamentals Part 2

Linux Machine Using SSH

SSH (Secure Shell) is a protocol that securely connects devices by encrypting data sent over a network, which is then decrypted on the remote machine.



- SSH allows us to remotely execute commands on another device remotely.
- Any data sent between the devices is encrypted when it is sent over a network such as the Internet

Flags and Switches

Most commands can take **arguments** called **flags** (or **switches)**. These start with a **hyphen (-)** and **adjust what the command does**.

By default, a command does its **normal** action. For example, `ls` shows the files in the current folder but **hides hidden files**. We can use **flags (switches)** to **change or extend** what a command does.

Using our `ls` example, `ls` informs us that there is only one folder named "folder1" as highlighted in the screenshot below.

Using ls to view the contents of a directory

```
tryhackme@linux2:~$ ls
folder1
tryhackme@linux2:~$
```

However, after using the `-a` argument (short for `--all`), we now have an output with a few more files and folders such as ".hidden folder". Files and folders with **"."** are hidden files.

Using ls to view hidden folders

```
tryhackme@linux2:~$ ls -a
.hiddenfolder folder1
tryhackme@linux2:~$
```

Commands that accept these will also have a `--help` option. This option will list the possible options that the command accepts.

Listing the options we can use with ls

```
tryhackme@linux2:~$ ls --help
Usage: ls [OPTION]... [FILE]...
List information about the FILEs (the current directory by default).
Sort entries alphabetically if none of -cftuvSUX nor --sort is specified.

Mandatory arguments to long options are mandatory for short options too.
  -a, --all                  do not ignore entries starting with .
  -A, --almost-all           do not list implied . and ..
      --author               with -l, print the author of each file
  -b, --escape               print C-style escapes for nongraphic characters
      --block-size=SIZE      with -l, scale sizes by SIZE when printing them;
                               e.g., '--block-size=M'; see SIZE format below
  -B, --ignore-backups       do not list implied entries ending with ~
  -c                         with -lt: sort by, and show, ctime (time of last
                               modification of file status information);
```

```
                             with -l: show ctime and sort by name;
                             otherwise: sort by ctime, newest first
  -C                         list entries by columns
      --color[=WHEN]         colorize the output; WHEN can be 'always' (default
                             if omitted), 'auto', or 'never'; more info below
  -d, --directory            list directories themselves, not their contents
  -D, --dired                generate output designed for Emacs' dired mode
  -f                         do not sort, enable -aU, disable -ls --color
  -F, --classify             append indicator (one of */=>@|) to entries
      --file-type            likewise, except do not append '*'
      --format=WORD          across -x, commas -m, horizontal -x, long -l,
                             single-column -1, verbose -l, vertical -C
      --full-time            like -l --time-style=full-iso
  -g                         like -l, but do not list owner
      --group-directories-first
tryhackme@linux2:~$
```

This option is, in fact, a formatted output of what is called the man page (short for manual), which contains documentation for Linux commands and applications.

The Man(ual) Page

The manual pages are a great source of information for both system commands and applications available on both a Linux machine, which is accessible on the machine itself and online.

To access this documentation, we can use the `man` command and then provide the command we want to read the documentation for. Using our ls example, we would use `man ls` to view the manual pages for `ls` like so:

Listing the options we can use with ls

Listing the options we can use with ls

```
tryhackme@linux2:~$ ls --help
Usage: ls [OPTION]... [FILE]...
List information about the FILEs (the current directory by default).
Sort entries alphabetically if none of -cftuvSUX nor --sort is specified.

Mandatory arguments to long options are mandatory for short options too.
  -a, --all                  do not ignore entries starting with .
  -A, --almost-all           do not list implied . and ..
      --author               with -l, print the author of each file
  -b, --escape               print C-style escapes for nongraphic characters
      --block-size=SIZE      with -l, scale sizes by SIZE when printing them;
                             e.g., '--block-size=M'; see SIZE format below
```

```
  -B, --ignore-backups       do not list implied entries ending with ~
  -c                         with -lt: sort by, and show, ctime (time of last
                               modification of file status information);
                               with -l: show ctime and sort by name;
                               otherwise: sort by ctime, newest first
  -C                         list entries by columns
      --color[=WHEN]         colorize the output; WHEN can be 'always' (default
                               if omitted), 'auto', or 'never'; more info below
  -d, --directory            list directories themselves, not their contents
  -D, --dired                generate output designed for Emacs' dired mode
  -f                         do not sort, enable -aU, disable -ls --color
  -F, --classify             append indicator (one of */=>@|) to entries
      --file-type            likewise, except do not append '*'
      --format=WORD          across -x, commas -m, horizontal -x, long -l,
                               single-column -1, verbose -l, vertical -C
      --full-time            like -l --time-style=full-iso
  -g                         like -l, but do not list owner
      --group-directories-first
tryhackme@linux2:~$
```

This option is, in fact, a formatted output of what is called the man page (short for manual), which contains documentation for Linux commands and applications.

The Manual Page

The manual pages are a great source of information for both system commands and applications available on both a Linux machine, which is accessible on the machine itself and online.

To access this documentation, we can use the `man` command and then provide the command we want to read the documentation for. Using our ls example, we would use `man ls` to view the manual pages for `ls` like so:

Listing the options we can use with ls

```
tryhackme@linux2:~$ man ls
LS(1)                                                   User Commands
LS(1)

NAME
       ls - list directory contents

SYNOPSIS
       ls [OPTION]... [FILE]...

DESCRIPTION
```

```
       List  information  about the FILEs (the current directory by default).  Sort
entries alphabetically if none of
       -cftuvSUX nor --sort is specified.

       Mandatory arguments to long options are mandatory for short options too.

       -a, --all
              do not ignore entries starting with .

       -A, --almost-all
              do not list implied . and ..

       --author
              with -l, print the author of each file

       -b, --escape
              print C-style escapes for nongraphic characters

       --block-size=SIZE
              with -l, scale sizes by SIZE when printing them; e.g., '--block-
size=M'; see SIZE format below

 Manual page ls(1) line 1 (press h for help or q to quit)
```

Filesystem Interaction

Command	Full Name	Purpose
touch	touch	Create file
mkdir	make directory	Create a folder
cp	copy	Copy a file or folder
mv	move	Move a file or folder
rm	remove	Remove a file or folder
file	file	Determine the type of a file

Creating Files and Folders (touch, mkdir)

You can create the file "note" by using `touch note`. It's worth noting that touch simply creates a blank file. You would need to use commands like echo or text editors such as nano to add content to the blank

file.

Using touch to create a new file

```
tryhackme@linux2:~$ touch note
tryhackme@linux2:~$ ls
folder1 note
```

This is a similar process for making a folder, which involves using the `mkdir` command and again providing the name that we want to assign to the directory. For example, creating the directory "my directory" using `mkdir mydirectory`.

Creating a new directory with mkdir

```
tryhackme@linux2:~$ mkdir mydirectory
tryhackme@linux2:~$ ls
folder1 mydirectory note
```

Removing Files and Folders (rm)

You can remove files by using `rm`. However, you need to provide the `-R` switch alongside the name of the directory you wish to remove.

Using rm to remove a file

```
tryhackme@linux2:~$ rm note
tryhackme@linux2:~$ ls
folder1 mydirectory
```

Using rm recursively to remove a directory

```
tryhackme@linux2:~$ rm -R mydirectory
tryhackme@linux2:~$ ls
folder1
```

Copying and Moving Files and Folders (cp, mv)

Copying and moving files is an important functionality on a Linux machine. Starting with `cp`, this command takes two arguments:

1. the name of the existing file

2. the name we wish to assign to the new file when copying

`cp` copies the entire contents of the existing file into the new file. In the screenshot below, we are copying "note" to "note2".

Using cp to copy a file

```
tryhackme@linux2:~$ cp note note2
tryhackme@linux2:~$ ls
folder1 note note2
```

Moving a file takes two arguments, just like the cp command. However, rather than copying and/or creating a new file, `mv` will merge or modify the second file that we provide as an argument. Not only can you use `mv` to move a file to a new folder, but you can also use `mv` to rename a file or folder. For example, in the screenshot we are renaming the file "note2" to be named "note3". "note3" will now have the contents of "note2".

Using mv to move a file

```
tryhackme@linux2:~$ mv note2 note3
tryhackme@linux2:~$ ls
folder1 note note3
```

Determining File Type

For example, we'll use `file` to confirm whether or not the "note" file in our examples is indeed a text file, like so `file note`.

Using file to determine the contents of a file

For example, we'll use `file` to confirm whether or not the "note" file in our examples is indeed a text file, like so `file note`.

Using file to determine the contents of a file

```
tryhackme@linux2:~$ file note
note: ASCII text
```

Permissions 101

Using ls -lh to list the permissions of all files in the directory

```
tryhackme@linux2:~$ ls -lh
-rw-r--r-- 1 cmnatic cmnatic 0 Feb 19 10:37 file1
-rw-r--r-- 8 cmnatic cmnatic 0 Feb 19 10:37 file2
```

These three columns are very important in determining certain characteristics of a file or folder and whether or not we have access to it. A file or folder can have a couple of characteristics that determine

both what actions are allowed and what user or group has the ability to perform the given action -- such as the following:

- Read
- Write
- Execute

The Differences Between Users & Groups

A user technically owns a file, if the permissions have been set, then a group of users can also have either the same or a different set of permissions to the same file without affecting the file owner itself.

For example the system user that runs a web server must have permissions to read and write files for an effective web application. However, companies such as web hosting companies will have to allow their customers to upload their own files for their website without being the webserver system user without compromising the security of every other customer.

Switching Between Users

Switching between users on a Linux install is easy work thanks to the `su` command. Unless you are the root user (or using root permissions through sudo), then you are required to know two things to facilitate this transition of user accounts:

- The user we wish to switch to
- The user's password

The `su` command takes a couple of switches that may be of relevance to you. For example, executing a command once you log in or specifying a specific shell to use. I encourage you to read the manual page for `su` to find out more. However, I will cover the `-l` or `--login` switch.

Simply, by providing the `-l` switch to `su`, we start a shell that is much more similar to the actual user logging into the system - we inherit a lot more properties of the new user, i.e., environment variables and the likes.

Using su to switch to user2 interactively

```
tryhackme@linux2:~$ su user2
Password:
user2@linux2:/home/tryhackme$
```

For example, when using `su` to switch to "user2", our new session drops us into our previous user's home directory.

Using su to switch to user2 interactively

```
tryhackme@linux2:~$ su -l user2
Password:
user2@linux2:~$ pwd
user2@:/home/user2$
```

Where now, after using `-l`, our new session has dropped us into the home directory of "user" automatically.

Common Directories

/etc

This root directory is one of the most important root directories on your system. The etc folder (short for etcetera) is a commonplace location to store system files that are used by your operating system.

For example, the sudoers file highlighted in the screenshot below contains a list of the users & groups that have permission to run sudo or a set of commands as the root user.

Also highlighted below are the "**passwd**" and "**shadow**" files. These two files are special for Linux as they show how your system stores the passwords for each user in encrypted formatting called sha512.

Some notable contents of the /etc directory

```
tryhackme@linux2:/etc$ ls
shadow passwd sudoers sudoers.d
```

/var

The "/var" directory, with "var" being short for variable data, is one of the main root folders found on a Linux install. This folder stores data that is frequently accessed or written by services or applications running on the system. For example, log files from running services and applications are written here (**/var/log**), or other data that is not necessarily associated with a specific user (i.e., databases and the like).

Some notable contents of the /var directory

```
tryhackme@linux2:/var$ ls
backups log opt tmp
```

/root

Unlike the **/home** directory, the **/root** folder is actually the home for the "root" system user. There isn't anything more to this folder other than just understanding that this is the home directory for the "root" user.

Some notable contents of the /root directory

```
root@linux2:~# ls
myfile myfolder passwords.xlsx
```

/tmp

This is a unique root directory found on a Linux install. Short for "temporary", the /tmp directory is volatile and is used to store data that is only needed to be accessed once or twice. Similar to the memory on your computer, once the computer is restarted, the contents of this folder are cleared out.

What's useful for us in pentesting is that any user can write to this folder by default. Meaning once we have access to a machine, it serves as a good place to store things like our enumeration scripts.

Some notable contents of the /tmp directory

```
root@linux2:/tmp# ls
todelete trash.txt rubbish.bin
```