Week 2: Introduction to Linux

6COSC019W

Dr. Ayman El Hajjar

Week 2

PLEASE READ THIS

- **NOTE**- In this lab, you only need a Linux machine. You can use the **Kali Linux machine**.
- THIS IS NOT MEANT TO BE A LAB TO FOLLOW STEP BY STEP. THIS IS A TUTORIAL ABOUT THE MAIN LINUX COMMANDS THAT WE WILL BE USING IN THIS MODULE AT SOME POINT.

PLEASE READ ALL OF THIS DOCUMENT

- Linux is more than an OS. It's an idea where everybody grows together and there's something for everybody.
- For this reason, there are many flavours and distributions for Linux.
- The two most commonly used architecture for those distributions are either RPM based or Debian based.
  - **RPM** Red Hat Linux and SUSE Linux were the original major distributions that used the .rpm file format, which is today used in several package management systems.
  - **Debian**: Debian is a distribution that emphasizes free software. It supports many hardware platforms. Debian and distributions based on it use the .deb package format[5] and the dpkg package manager and its frontends (such as apt-get or synaptic)
- Kali and Ubuntu, are Debian. In this lab we are using Kali Linux.

A note

- In most cases, Linux errors are self explanatory. Read the error the terminal throws at you.

1 Learning the shell

- When we speak of the command line, we are really referring to the shell. The shell is a program that takes keyboard commands and passes them to the operating system to carry out.
- Almost all Linux distributions supply a shell program from the GNU Project called bash.
- When using a graphical user interface, we need another program called a terminal emulator to interact with the shell.
- It is called **terminal**

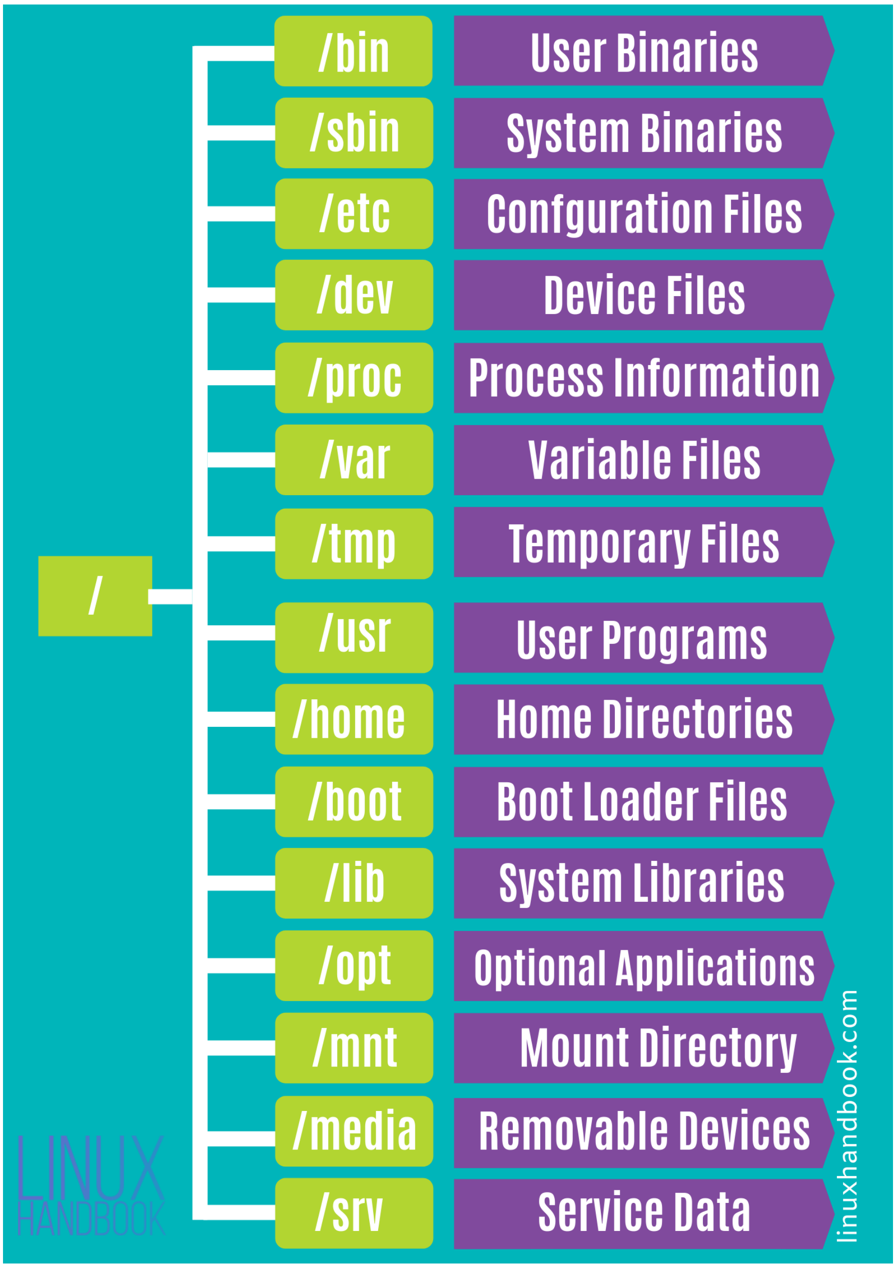
1.1 Simple Commands

- **date**: This command displays the current time and date.
- **df**: To see the current amount of free space on your disk drives
- **free**: Likewise, to display the amount of free memory.
- If we press the up-arrow key we can see previous commands.
- **history**: This command remembers the last 500 commands by default in most Linux distributions.
- We can end a terminal session by either closing the terminal emulator window or entering the **exit** command at the shell prompt.

1.2 File system tree

- Like Windows, a Unix-like operating system such as Linux organizes its files in what is called a hierarchical directory structure.
- This means that they are organized in a tree-like pattern of directories (sometimes called folders in other systems), which may contain files and other directories.
- The first directory in the filesystem is called the root directory. The root directory contains files and subdirectories, which contain more files and subdirectories, and so on.

- Note that unlike Windows, which has a separate filesystem tree for each storage device, Unix-like systems such as Linux always have a single filesystem tree, regardless of how many drives or storage devices are attached to the computer.
- Storage devices are attached (or more correctly, mounted) at various points on the tree according to the whims of the system administrator, the person (or persons) responsible for the maintenance of the system.



All files on a Linux system are stored on file systems which are organized into a single inverted tree of directories, know as a file system hierarchy.In the inverted tree, root lies at the top and the branches of directories and sub-directories stretch below the root.

- \usr
  - Locally customized software
- \bin
  - User commands.
- \sbin
  - System administration commands.
- \etc
  - Configuration files specific to this system.
- \etc
  - Configuration files specific to this system.
- \var
  - Files that dynamically change (e.g. databases, cache directories, log files, printer spooled documents, and website content) may be found under /var.
- \home
  - Home directories where regular users store their personal data and configuration files.
- \root
  - Purpose
    - Home directory for the administrative superuser, root.
- \tmp
  - Purpose
    - A world-writable space for temporary files. Files more than 10 days are automatically be deleted from that directory.

1.3 Commands to navigate file systems

Commands options

- Most commands use options consisting of a single character preceded by a dash, such as -l.
- But many commands, including those from the GNU Project, also support long options, consisting of a word preceded by two dashes.
- If you want to know what options you can use for specific commands, you can read the manual for each command.
- **man**: If you type **man ls** for example, this will give you the manual for ls command including all options for ls.

- **pwd**: This command displays the current working directory.
  - When you are using GIU, you can easily identify in which directory you are working.
  - When you using command line, you need to use a command to help you identify this.
- **ls**: lists the files and directories in the current working. directory.
  - **ls -a** or **ls --all** List all files, even those with names that begin with a period, which are normally not listed (i.e., hidden).
  - **-d --directory** Changes the working directory to the previous working directory.
  - **-F --classify** This option will append an indicator character to the end of each listed name (for example, a forward slash if the name is a directory).
  - **-h --human-readable** In long format listings, display file sizes in human-readable format rather than in bytes.
  - **-l** display its results in long format. This format contains a great deal of useful information.
- **cd**: To change your working directory (where we are standing in our tree-shaped maze) we use the **cd** command: Type cd followed by the pathname of the desired working directory
    * **cd /bin**
  - This will change your working directory to **bin** directory
  - There are some shortcuts for the cd command
    * **cd** Changes the working directory to your home directory.

    * **cd -** Changes the working directory to the previous working directory.
    * **cd ..** move to the parent of the current working directory.
    * **cd** ~ Changes the working directory to the home directory of username.

* **file**: We will use the file command to determine a file's type.

* **less**: The less command is a program to view text files.
  - Once started, the less program allows you to scroll forward and backward through a text file. For example, to examine the file that defines all the system's user accounts, enter the following command:
    * **less /etc/passwd**

2 Checking Your Login with whoami

- If you've forgotten whether you're logged in as root or another user, you can use the whoami command to see which user you're logged in as:
  * **whoami**

3 Manipulating Files and Directories

* **mkdir**: The mkdir command is used to create directories.
  * **mkdir lab1** will create a directory called lab1
  * type **man mkdir** to learn about the different options of the **mkdir**

- **cp**: The cp command copies files or directories.
  * **cp item1 item2** will copy the single file or directory item1 to file or directory item2
  * type **man cp** to learn about the different options of the **cp** command

- **mv**: The mv command performs both file moving and file renaming, depending on how it is used.
  - **mv item1 item2** move or rename file or directory item1 to item2
  - type **man mv** to learn about the different options of the **mv** command

- **rm**: The rm command is used to remove (delete) files and directories.
  - **rm item1** will remove item1 file or directory item1 (if empty)
  - type **man rm** to learn about the different options of the **rm** command and how to delete a directory even if it contains documents.

3.1 Text manipulation

- Viewing Files: You can use **cat** command to display a text file in the terminal window. For example,ls if we want to read the contents of the configuration file for adduser , to display its contents in the terminal, we type:
  - **cat /etc/adduser.conf**
- If the file is too long, **cat** command, you might not see the whole document on your terminal window. You can use instead **head** or **tail** commands to display parts of the file's content in order to more easily view the key content.
  - Use head to show the first few lines of the document
    - **head /etc/adduser.conf**
  - Use head and the number of lines to show a specific number of lines of the document. For example the **-20** option will show the first 20 lines.
    - **head -20 /etc/adduser.conf**
  - The tail command is similar to the head command, but it's used to view the last lines of a file. Let's use it on adduser.conf:
    - **tail /etc/adduser.conf**
  - Use tail and the number of lines to show a specific number of lines of the document. For example the **-20** option will show the last 20 lines.
    - **tail -20 /etc/adduser.conf**
- To display file and numbering the lines, use the command **nl**.
  - **nl /etc/adduser.conf**
- Show text directly on terminal: You can use **echo** command to display a text file in the terminal window. For example, if we want to display the text "this is my text" in the terminal, we type:
  - **echo "this is my text"**
- The command **touch** is used to create empty files or update the timestamps of existing files.
  - **touch** Manipulate file timestamps, either by creating new empty files or by updating the access and modification times of existing files.
  - For example
    - to create an empty file
      - touch filename
    - If the file **filename** exists and we use touch on it
      - touch will simply update the timestamp
    - you can also use it to create several files at the same time

            - touch file1 file2 file3
    - manual of touch contains many more options
        - man touch

- Suppose we want to add the text "This is my first line" to the end of the file named "example". We use:
    - Make sure you are on Desktop. Type cd ~/Desktop/
    - then we type:

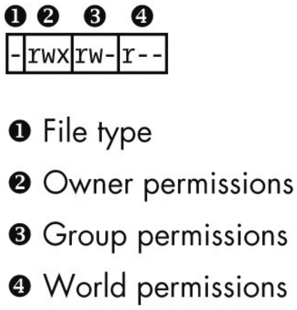
      echo 'This is my first line' » example

        - This will add the line **This is my first line** to the end of the example file. How?
            - **echo 'This is my first line'**: This part of the command prints the text "root" to the standard output (usually the terminal).
            - **» example**: The » operator is used for output redirection. It appends the output of the preceding command to the specified file, which, in this case, is "example". If the file doesn't exist, it will be created. If the file already exists, the text "This is my first line" will be added as a new line at the end of the file.

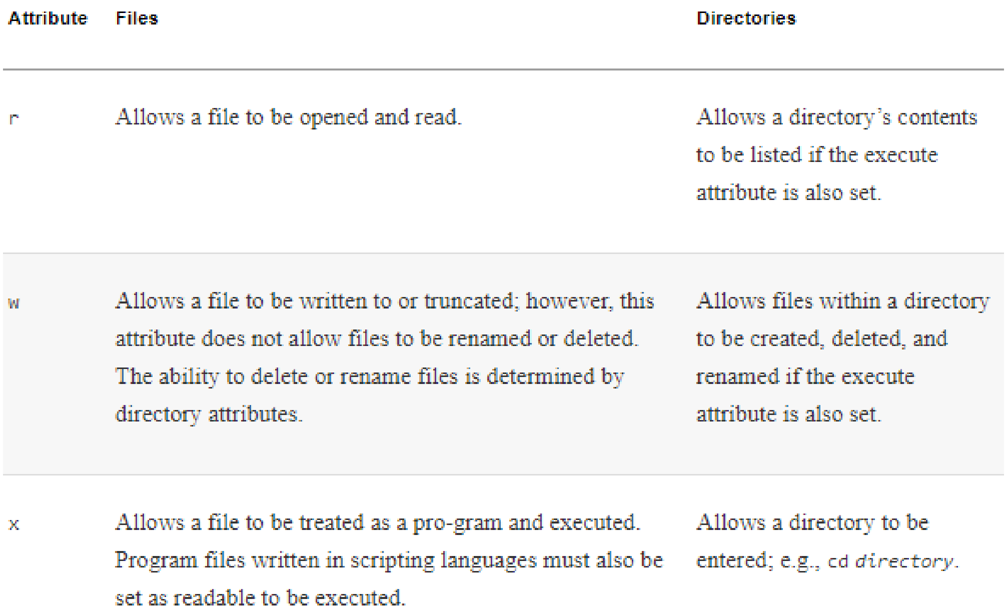
4 Permission

- Operating systems in the Unix tradition differ from those in the MS-DOS tradition in that they are not only multitasking systems but also multiuser systems.
- What exactly does this mean? It means that more than one person can use the computer at the same time. While a typical computer will likely have only one keyboard and monitor, it can still be used by more than one user. For example, if a computer is attached to a network or the Internet, remote users can log in via ssh (secure shell) and operate the computer. In fact, remote users can execute graphical applications and have the graphical output appear on a remote display. The X Window System supports this as part of its basic design.
- The multiuser capability of Linux is not a recent "innovation" but rather a feature that is deeply embedded into the design of the operating system. Considering the environment in which Unix was created, this makes perfect sense. Years ago, before computers were "personal," they were large, expensive, and centralized. A typical university computer system, for example, consisted of a large central computer located in one building and terminals located throughout the campus, each connected to the large central computer. The computer would support many users at the same time.

- In order to make this practical, a method had to be devised to protect the users from each other. After all, the actions of one user could not be allowed to crash the computer, nor could one user interfere with the files belonging to another user
- Access rights to files and directories are defined in terms of read access, write access, and execution access. If we look at the output of the ls command, we can get some clue as to how this is implemented.

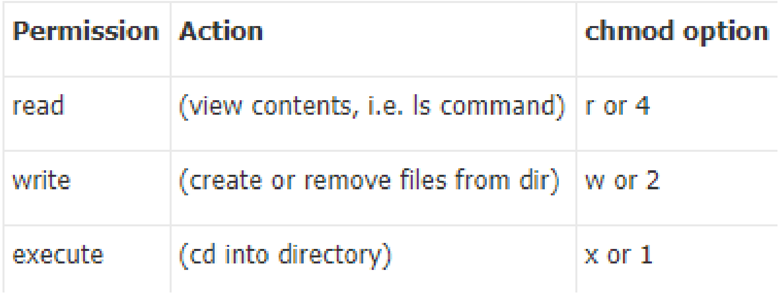


- The remaining nine characters of the file attributes, called the file mode, represent the read, write, and execute permissions for the file's owner, the file's group owner, and everybody else.
- When set, the r, w, and x mode attributes have certain effects on files and directories as below and using **chmod** command.



Attribute	Files	Directories
r	Allows a file to be opened and read.	Allows a directory's contents to be listed if the execute attribute is also set.
w	Allows a file to be written to or truncated; however, this attribute does not allow files to be renamed or deleted. The ability to delete or rename files is determined by directory attributes.	Allows files within a directory to be created, deleted, and renamed if the execute attribute is also set.
x	Allows a file to be treated as a pro-gram and executed. Program files written in scripting languages must also be set as readable to be executed.	Allows a directory to be entered; e.g., `cd directory`.

- To change the mode (permissions) of a file or directory, the chmod command is used. Be aware that only the file's owner or the superuser can change the mode of a file or directory. chmod supports two distinct ways of specifying mode changes: octal number representation and symbolic representation. We will cover octal number representation first.
  - Directories have directory permissions. The directory permissions restrict different actions than with files or device nodes.



Permission	Action	chmod option
read	(view contents, i.e. ls command)	r or 4
write	(create or remove files from dir)	w or 2
execute	(cd into directory)	x or 1

- To give file permission for a file called filename for example in this specific order execute for owner , read & write for group and read for other, we use the command as below:
    - * **chmod 764 filename**
  - Where 7 is read write execute $4 + 2 + 1 = 7$
  - Where 6 is read write $4 + 2 = 6$
  - Where 4 is read 4

Super user commands

- One of the recurrent problems for regular users is how to perform certain tasks that require superuser privileges.

- These tasks include installing and updating software, editing system configuration files, and accessing devices.

- In the Windows world, this is often done by giving users administrative privileges. This allows users to perform these tasks. However, it also enables programs executed by the user to have the same abilities. This is desirable in most cases, but it also permits malware (malicious software) such s viruses to have free run of the computer.

- To be able to use commands that requires higher privilege you should use **sudo** command

- You will be asked for a password.

  - In linux when you enter a password in the terminal, nothing shows up. This is done on purpose in order to hide the length of the password.
  - If for example I want to copy a file called run.sh to bin directory, I will need root privilege. To do this I type
    * **sudo cp run.sh /bin**

- **Read this** You should not use **su** command. su command give you full root privilege for everything you do afterwards and do not ask you for a password. This is dangerous and you can break your system if you do something wrong. It is better to only use **sudo**. **sudo** which stands to **super use do** will remind you all the time to enter the password. This gives you chance to stop the command.

Networking

- We need to know the IP address of our machine.
- You can use **ifconfig** command or **ip address** or **ip a** command to do so.
  - This will list all your network interfaces and the different IP address allocated for each interface. Interfaces can be for example a network card, a wireless network card, or a virtual network card.
  - if ifconfig doesn't work and you get an error to install the command. you can type **sudo apt-get install net-tools** to install it. We will explain this command later.
- If I want to bring down a specific interface (net1 for example) I can type
    * **ifconfig net1 down**
- If I want to bring up a specific interface (net1 for example) I can type
    * **ifconfig net1 up**
- If your interface was not allocated an IP address dynamically you can set it up yourself. For example if I want to give IP 192.168.56.102 for the interface net1 I use the command below. Not the netmask is your subnet address.
    * **ifconfig net1 192.169.56.10 netmask 255.255.255.0**
- If I want to check connectivity with another device on my network I use the command **ping**
    * **ping 192.168.56.103** will try to connect using ICMP protocol with this ip and see if this IP is reachable.
- If I want to request an IP address from your the DHCP server, you can use dhclient command. For example if my interface is called eth0 I use the command below to assign it an IP address dynamically.
    * **sudo dhclient eth0**

- If you want to check if your wireless interface is up, or if it is plugged in, you use the command **iwconfig**. iwconfig will show on the terminal which interface has wireless capabilities.

  * **iwconfig**

5 Adding and removing software

- When doing your labs, sometimes you will have to add or install some software.
- You might even want to remove some other software.
- In Linux, the default software manager is called Advanced Packaging tool or **apt**. You can use either **apt** or **apt-get** for this command but apt-get has more functionality.
- To use apt, you can easily specify what do you want to do with apt. For example to install an graphic text editor called **mousepad**, similar to **notepad** in windows. We can install it by simply typing:

  * **sudo apt-get install mousepad**

- Install **net-tools** if ifconfig was not available in the previous task.

  * **sudo apt-get install net-tools**

- you can replace gedit by the name of the software you want to install. The software need to exist in Linux packages repository.
- You can also remove the software you just installed by typing:

  * **sudo apt-get remove chromium**

- Remove will only remove a software without removing its configuration file. If you want to remove the software and the configuration file, you should use the **purge** option

  * **sudo apt-get purge chromium**

- The software repository list is updated regularly. On your system you should update the repository to download the latest repository links. You can do this by typing:

  * **sudo apt-get update**

- You can also edit the repository list manually, if you want to add a software to your list that is not added by default to your Linux repository.
- We have already updated the repository in the lab environment document to fix an error you get when you are in the university premises.

- The apt source file is called sources.list and is in location **/etc/apt/**
- You can edit the file using any text editor you prefer to use. There are many that have a graphical interface. Some are only terminal based.
- To edit it using a graphical based editor:
  - * **sudo mousepad /etc/apt/sources.list**
- To edit it using a terminal based editor:
  - * **sudo nano /etc/apt/sources.list**
- You can also install software that are on github. To do this you will need first to install the git software using apt. You can type:
  - * **sudo apt-get install git**
- Once you have the **git** software installed, you can then download scripts and software directly from github. For example, to download SIGT, a nice and easy information gathering toolkit that you can use for our next week lab, you type:
  - * **sudo git clone https://github.com/termuxhackers-id/SIGIT.git**

6 Pipes and connecting things

- Pipes Take the output of one command and send it to another
- For example we can take output of more reading a string of text and count
  - * **echo "this is my first Linux lab"| wc**
    - ∗ In this example, the terminal executes the first part of the command, but the output is redirected as an input for the second part of the command after the pipe.
    - ∗ **echo** prints the specified text
    - ∗ **wc** Stands for "word count." It's a command that counts the number of lines, words, and characters in its input.
    - ∗ | The pipe operator. It takes the output of the command on its left and uses it as the input for the command on its right.
  - – pipes become really important and useful when you are want the output of certain commands to be used as input for others.
- In Linux, a lot of what we'll be working with at the command line involves text files or text output
- Now let's see how we can search for files and streams.
- The grep command searches files for matching patterns. This io what streams is.

- For example, I need to see if file1.txt contains a specific word, let's say the word "Linux". I type:
    * **grep "Linux" file1.txt**
    - In this case grep returned the lines that contains the string "Linux"
    - If you get many results, it might be difficult to make use of them if you are trying to debug code for example. In this case, you might want to add the line number to the output so you can find it easily. To do so, you type:
    * **grep -n "Linux" file1.txt**

- The command grep is probably the most widely used text manipulation command. It lets you filter the content of a file for display. If, for instance, you want to see all lines that include the word groups in your adduser.conf file, you could use cat and ask it to display only those lines.
    * **cat /etc/adduser.conf | grep groups**

- To make use of the data and logs if you are a cyber security analyst, you need to find how to look at files and evaluate them quickly, specially if you are dealing with hundreds or thousands of logs.

- Let's see how we can find and extract things quickly

- One really useful command in Linux is **awk**

- **awk** is used to extract specific text from a file according to a specific rule. You can create an awk program (or rule) either at the command line or store it in a file so you can run it when you need to.

- Download the log file users.log from blackboard. This log file contains usernames and how strong their passwords are. The delimiters is a "tab" which is separating the columns in my data.

- Let's say I want to search for all the data in the second column.
    * **awk '{print $2}' example.txt**
    - note: awk is an extremely powerful command that allows you to write rules in anyway you want. I would encourage you to explore it further if you want to develop your skills in Linux.