Chapter 4 in Class Assignment
ITN 261
Bradyn Ritchie

Laboratory Exercise ITN-261 In-class – Introduction to Linux User Accounts, Groups, and Permissions.

1. Overview

This lesson will provide an introduction into Linux user accounts. By the end of this lesson, you should be comfortable creating, deleting, and managing, user accounts. Groups and Permissions will also be introduced.

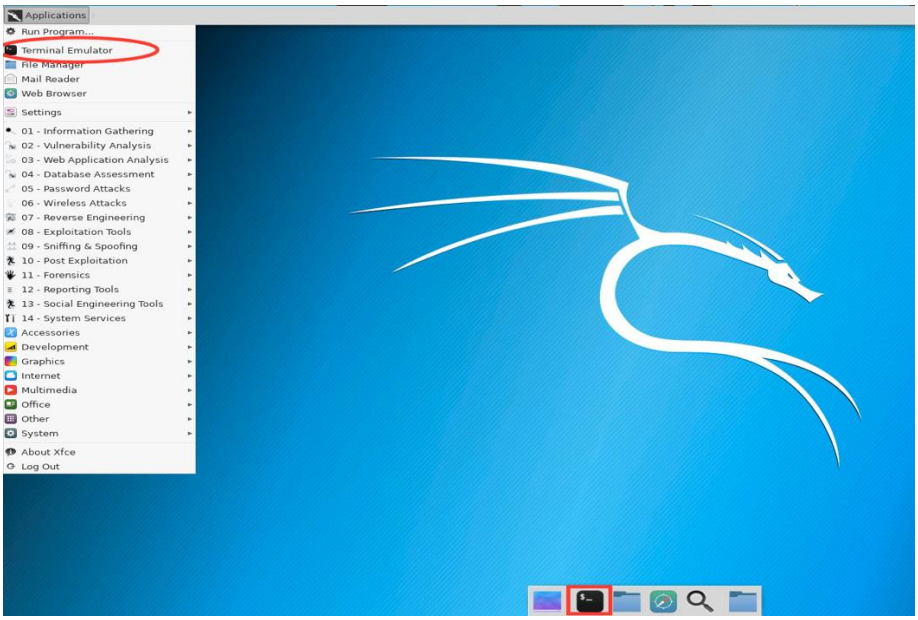
2. Resources required

Virginia Cyber Range course with Kali Linux Virtual Machine exercise environment OR Cyber Basics (2018) exercise environment.

3. Initial Setup

This exercise requires a VirtualBox Kali Linux Virtual Machine running on a computer (laptop) or a Kali Linux VM running in the Virginia Cyber Range. If using Virginia Cyber Range account, login to your Kali Linux virtual machine and launch a terminal window. You can launch a terminal window either by accessing it through the **Applications** menu at the top right and selecting **Terminal Emulator** or clicking the black terminal icon at the bottom of the screen. See image below.

Username: student
Password: student



4. Tasks

VIRGINIA CYBER RANGE

Chapter 4 in Class Assignment
ITN 261
Bradyn Ritchie

Task 1: Adding a user

In order to add a user, we will use the "`adduser`" command. Let's go ahead and create a user by typing the following:

```
$ sudo adduser john
```

[NOTE: This *may* require you to enter your account password to create the account, but should not in the Virginia Cyber Range environment. This is because we are using the `sudo` command. The `sudo` command means "super user do" which allows a user, with the proper permissions, to perform tasks at an elevated level of permission. This command is used to execute higher-privileged tasks and may require a password. If prompted for a password, enter the "student" account password which is "student."]

The program will now prompt you for a password for john, simply enter "john" as the password for the account. NOTE: Linux will not display the password or any other indication that you are typing anything. Don't be fooled, this is done for security reasons.

```
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
Changing the user information for john
Enter the new value, or press ENTER for the default
```

Once you get to this point, you will be prompted for the following:

```
        Full Name []:
        Room Number []:
        Work Phone []:
        Home Phone []:
        Other []:
Is the information correct? [Y/n]
```

You do not have to enter any information, simply hit enter until you get back to the command prompt.

IMPORTANT NOTE: There are two programs to create a user: `adduser` and `useradd`.

- `useradd` is a native binary that will create a user account with no password.
- `adduser` is a perl script that utilizes the native binary useradd and will create the account with more specifics (i.e. password).

A general rule of thumb is to use `adduser` to create the account with a password immediately. If you wish, however, to create a user with the `useradd` command…then do the following:

```
$ sudo useradd alice
$ sudo passwd alice
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
```

```
$
```

Commands:

`sudo` (super user do) – a program to allow certain, authorized, users to perform sensitive actions.
`useradd` – create a new user account with no other specifications.
`adduser` – create a user account with a password and other information (uses useradd).
`passwd` – change the password of a user.

Task 2: Login as different user

Now for us to use the new account we just created. Type the following at the command prompt:

```
$ whoami
student
$ sudo login john
```
Enter password for john (john)
```
$ whoami
```

Notice that `whoami` provides you with the name of the current logged in account. There are two ways you can logout of an account when you are done:

```
$ logout
OR
$ exit
```

NOTE: `exit` will close the terminal when you are back to the original user.

Task 3: Delete user accounts

When it comes to deleting an account, you can do so with the `userdel` command. Type the following:

```
$ sudo userdel john
```

The user should have been removed from the system after executing this command.

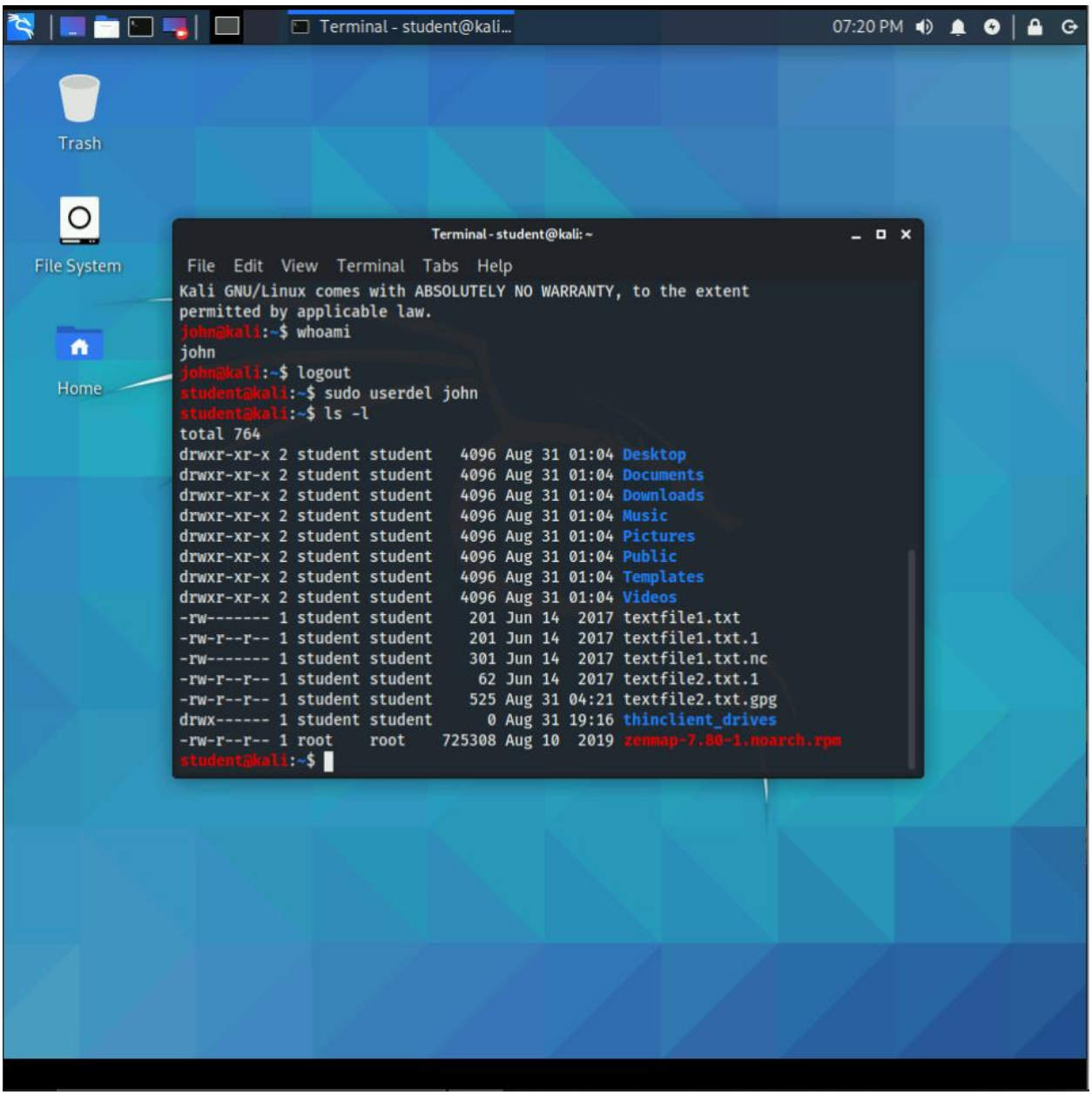
Task 4: Understanding user permissions

In this task we are going to explore a rather important concept – permissions. Permissions are a very vital part to any Linux systems administration and security. Let's revisit a command we have seen before to explore permissions. Run the following command in the /home/student directory:

```
$ ls -l
OR
$ ls -l /home/student
```

Chapter 4 in Class Assignment
ITN 261
Bradyn Ritchie

INSERT Screenshot here:



We have used this command before to distinguish between files and directories, but now we will learn more about the output. You will notice a string of text such as the following:

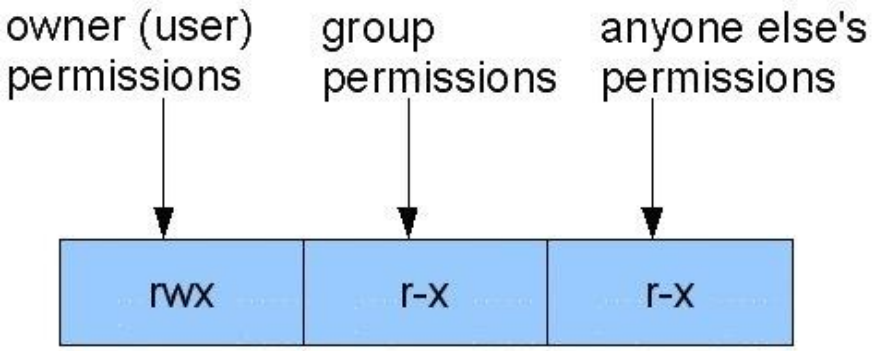
```
drwx-wx—x
d-wxrwxrwx
d—x—xrwx
```

As we know, the d represents a directory, but what about the other text? Simply put…

r = read
w = write
x = execute

There are groups of three, which are to separate the permissions between three groups of users:

1. Owner
2. Groups
3. Other



(Source: https://s11986.pcdn.co/wp-content/uploads/2013/03/linux-permissions.jpg)

Below, we will explore groups more thoroughly.

Task 5: Understanding groups

In the Linux system, every user is placed into groups. These groupings are meant to make logical organization of the users in the system. This is done to separate users for segregation of tasks and permissions. Let's, for example, create three accounts. We will use `useradd` for simplicity sake, but remember to use `adduser` as a rule of thumb. Type the following to create the three accounts bob, billy, and james and the chess_club group:

```
$ sudo useradd bob
$ sudo useradd billy
$ sudo useradd james
$ sudo groupadd chess_club
$ sudo usermod -a -G chess_club bob
$ sudo usermod -a -G chess_club billy
$ groups billy
$ groups bob
$ groups james
```

You should have observed that billy and bob were both added to the chess_club group we created but james was not. This will allow certain content to be created to allow the billy and bob accounts to access the chess_club data and prevent the account james, who is not a member, from viewing the data.

Commands:

`groupadd` – Create a new group in the system.

`usermod` – Make a change to a user account. Our example above uses flags to add a user to a specific group.
`groups` – View what groups the user is a part of.

Task 6: Changing permissions

Lastly, we are going to take a look at how to change permissions for a specific file or folder. We can do this with the "`chmod`" command. At a command prompt, type the following:

```
$ touch new_file
$ ls -l new_file
$ chmod 722 new_file
$ ls -l new_file
```

To reiterate, the `touch` command will update the timestamp of an existing file or create a new file if a file of that name does not already exist.

You will notice that the permissions have changed. The owner can read write and execute, and groups and others can only write. Now let's learn the secret behind the number notation of "722."

Each of the three permitted actions (read, write, and execute) are each denoted by their own number.

r – 4
w – 2
x – 1

So, this explains why we ended up with writing permissions for groups and others, but what about the owner having all three? Notice, 4+2+1 = 7; therefore, you are able to perform any combination to yield a certain permission set you wish. For example, if you wanted the owner to have read and write permissions, and give no permissions to groups and others, then r=4, w=2, 4+2=6. Now at the terminal, type the following command.

```
$ chmod 644 new_file
```

What permissions did you just grant the new file? Using 4, 2, and 1, we can check our math.

6 = 4 + 2 = r + w
and
4 = 4 = r

So, the owner of the new file has read and write privileges, and groups and others just have read permission.

NOTE: It is NEVER recommended that you perform "chmod 777" on ANYTHING. You should never grant permissions to everyone to do everything in a system!

Chapter 4 in Class Assignment
ITN 261
Bradyn Ritchie

QUESTIONS:

1. **Which is more recommended you run to add a user? `useradd` or `adduser`? Why?**

Adduser should always be used over useradd. This is because adduser, upon the creation of a new user, allows you toad more information and do numerous things with the account. Using adduser automatically creates the account's home folders and allows you to be able to set up a password for the newly created account. When creating a new user with the useradd command, it only creates a new user under the name of what you input. This would really only be best used if there was a temporary user on the system. No folders or passwords are created for an account when using the useradd command.

2. **What is `sudo`? Why is it required for most of the commands we are performing?**

Sudo, meaning "super user do", is a command that allows certain, authorized user to perform sensitive actions. It essentially allows us to run a program or do a task as the root user. This is required for most of the commands we are performing because, with our normal student account, we do not have the proper permissions to create or delete users/groups. Trying to run the commands without sudo would result in the error "only root may add a user or group to the system". Using sudo essentially elevates us to the position of root user, allowing us to modify the users and groups on the system.

3. **Let's assume you have a folder named "stuff" and you wish to make "stuff" readable and writable for the owner, readable by groups and others, how would you do this?**

You would want to run the command "chmod 644 stuff". Since the permissions correspond to difference numbers, readable and writable for the owner would be 4 + 2 = 6 and readable for the other two would be 2 since the value of readable permissions is 2.

4. **Interpret these permissions: `dr-xrw--w-`**
   a. **Owner:** Read, execute 5
   b. **Group:** Read, write 6
   c. **Other:** Write 2

The permissions are divided up into sets of three corresponding to owner, group, and other permissions. Owner only has r and x, meaning read and execute. Group has r and w, meaning read and write. Other only has w, meaning write. The numerical value for these would be "562".