Portfolio: Linux Lab :

Link to videos Recordings:

LINUX:
https://drive.google.com/drive/folders/10l66EblVzahm8ITEd6dWs4Mu5Msj2hZW?usp=sharing

LOG Analysis:
https://drive.google.com/drive/folders/1LEuEFPwyLwPDtSlDcxy4lxlzR5OIPVR4?usp=sharing

Install software in a Linux distribution

Activity overview

In this lab activity, you'll use the Advanced Package Tool (APT) and `sudo` to install and uninstall applications in a Linux Bash shell.

While installing Linux applications can be a complex task, the APT package manager manages most of this complexity for you and allows you to quickly and reliably manage the applications in a Linux environment.

You'll use Suricata and tcpdump as an example. These are network security applications that can be used to capture and analyze network traffic.

The virtual machine you access in this lab has a Debian-based distribution of Linux running, and that works with the APT package manager. Using a virtual machine prevents damage to a system in the event its tools are used improperly. It also gives you the ability to revert to a previous state.

As a security analyst, it's likely you'll need to know how to install and manage applications on a Linux operating system. In this lab activity, you'll learn how to do exactly that!

Scenario

Your role as a security analyst requires that you have the Suricata and tcpdump network security applications installed on your system.

In this scenario, you have to install, uninstall, and reinstall these applications on your Linux Bash shell. You also need to confirm that you've installed them correctly.

Here's how you'll do this: **First**, you'll confirm that APT is installed on your Linux Bash shell. **Next**, you'll use APT to install the Suricata application and confirm that it is installed. **Then**, you'll uninstall the Suricata application and confirm this as well. **Next**,

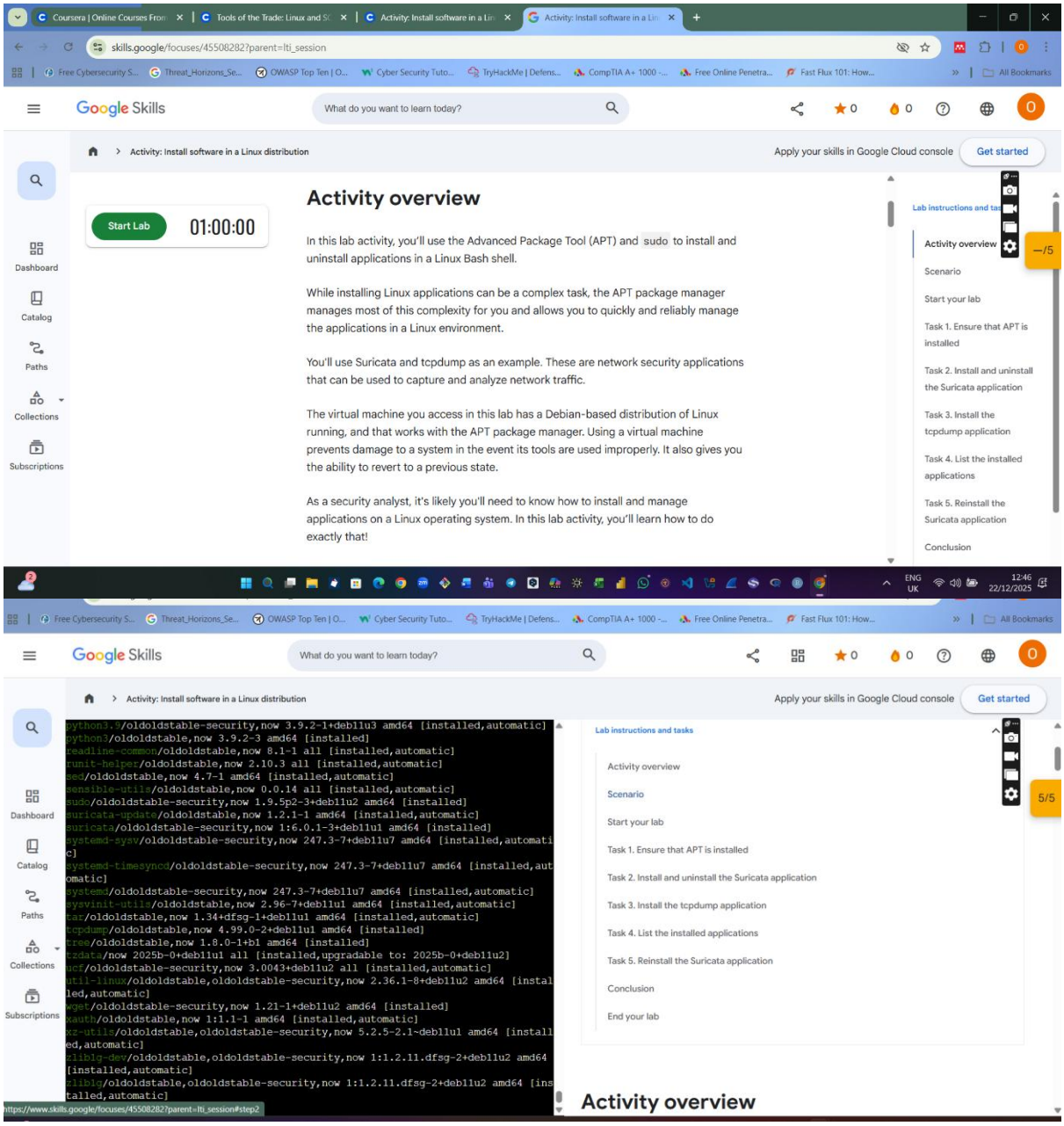
you'll install the tcpdump application and list the applications currently installed. **Finally**, you'll reinstall the Suricata application and confirm that both applications are installed.

Lab practice 1: on the use of apt package tool (Advanced Package Tool); suricata and tcpdump installation and uninstalling.

```
apt
sudo apt install suricata
suricata
sudo apt remove suricata
sudo apt install tcpdump
tcpdump
apt list –installed
```

Output:

```
...
suricata/oldstable,now 1:4.1.2-2+deb10u1 amd64 [installed]
...
tcpdump/oldstable,now 4.9.3-1~deb10u2 amd64 [installed]
...
```



#

Video recording: https://screenrec.com/share/NkeE5ihLmR

2. Activity: Examine input and output in the shell

Introduction

In this lab, you'll use the **`echo`** command to examine how input is received and how output is returned in the shell. You'll also use other Linux commands in the Bash shell to explore more about input and output and other basic functions of the shell

What you'll do

You have multiple tasks in this lab:

- Generate output in the shell the **`echo`** command
- Perform basic calculations the **`expr`** command
- Clear the shell window the **`clear`** command
- Explore the commands further

Lab practice 2: on the Use of Bash

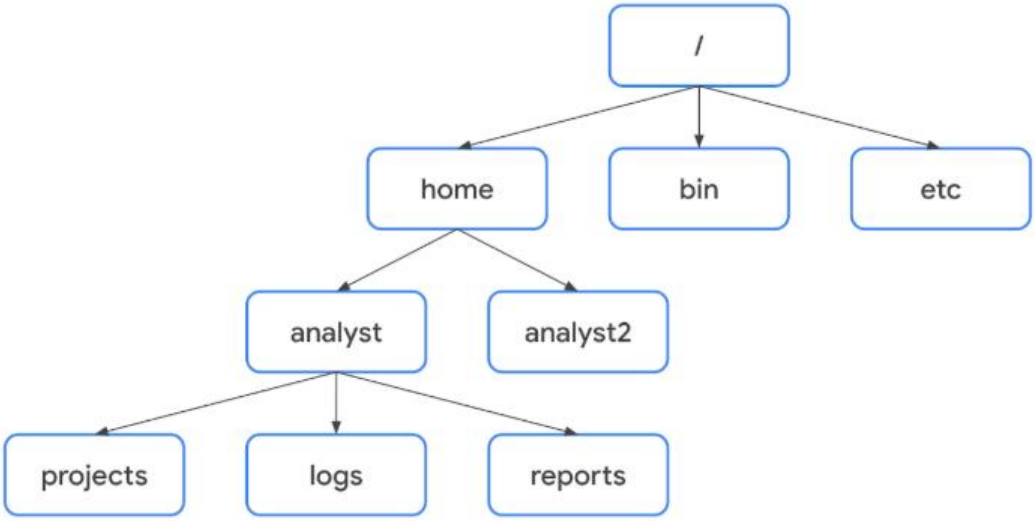
```
echo osita
echo “osita”
expr 3 + 4
clear
```

Lab practice 3: Commands Used in Linus

Pwd – current working directory. The absolute file path. The **`whoami`** = username

```
Ls - list
Cd – change directory
cat (file name)– displays the content of a file
head (file name) – content display by default 10 lines of the file
```

Filesystem Hierarchy Standard



- A **file path** is the location of a file or directory
- **root directory -** (`/`)
- `/home`: Each user in the system gets their own home directory.
- `/bin`: This directory stands for “binary” and contains binary files and other executables. Executables are files that contain a series of commands a computer needs to follow to run programs and perform other functions.
- `/etc`: This directory stores the system’s configuration files.
- `/tmp`: This directory stores many temporary files. The `/tmp` directory is commonly used by attackers because anyone in the system can modify data in these files.
- `/mnt`: This directory stands for “mount” and stores media, such as USB drives and hard drives.

Pro Tip 1: You can use the `man hier` command to learn more about the FHS and its standard directories.

Pro Tip 2: When the path leads to a subdirectory below the user’s home directory, the user’s home directory can be represented as the tilde (`~`). For example, `/home/analyst/logs` can also be represented as `~/logs`.

Pro Tip 3: Relative file paths can use a dot (`.`) to represent the current directory, or two dots (`..`) to represent the parent of the current directory. An example of a relative file path could be `../projects`.

Pro Tip 4: The **absolute file path** is the full file path, which starts from the root. For example, `/home/analyst/projects` is an absolute file path

Note: If you want to return the contents of a directory that's not your current working directory, you can add an argument after `ls` with the absolute or relative file path to the desired directory. For example, if you're in the `/home/analyst` directory but want to list the contents of its `projects` subdirectory, you can enter `ls /home/analyst/projects` or just `ls projects`.

if you're in the `/home/analyst` directory and want to navigate to its `projects` subdirectory, you can enter `cd projects`.

if you're in `/home/analyst/projects`, entering `cd /home/analyst/logs` changes your current directory to `/home/analyst/logs`.

ou can use the relative file path and enter `cd ..` to go up one level in the file structure. For example, if the current directory is `/home/analyst/projects`, entering `cd ..` would change your working directory to `/home/analyst`.

Reading File Content: `cat`, `head`, `tail`, and `less`.

`Cat:` entering `cat updates.txt` returns everything in the `updates.txt` file.

Head : displays the content of the file

if you only want to display the first five lines of the `updates.txt` file, enter `head -n 5 updates.txt`.

tail: The `tail` command does the opposite of `head`. This command can be used to display just the end of a file, by default 10 lines. Entering `tail updates.txt` returns only the last 10 lines of the `updates.txt` file.

Pro Tip: You can use `tail` to read the most recent information in a log file.

Less: The `less` command returns the content of a file one page at a time.

Once you've accessed your content with the `less` command, you can use several keyboard controls to move through the file:

- `Space bar`: Move forward one page
- `b`: Move back one page
- `Down arrow`: Move forward one line
- `Up arrow`: Move back one line

- `q`: Quit and return to the previous terminal window

Lab Practice 4:

Activity overview

Previously, you learned about Linux and how to communicate with the OS through the shell. You also learned how to use some of the core commands to navigate the Linux file system and read content from files it contains.

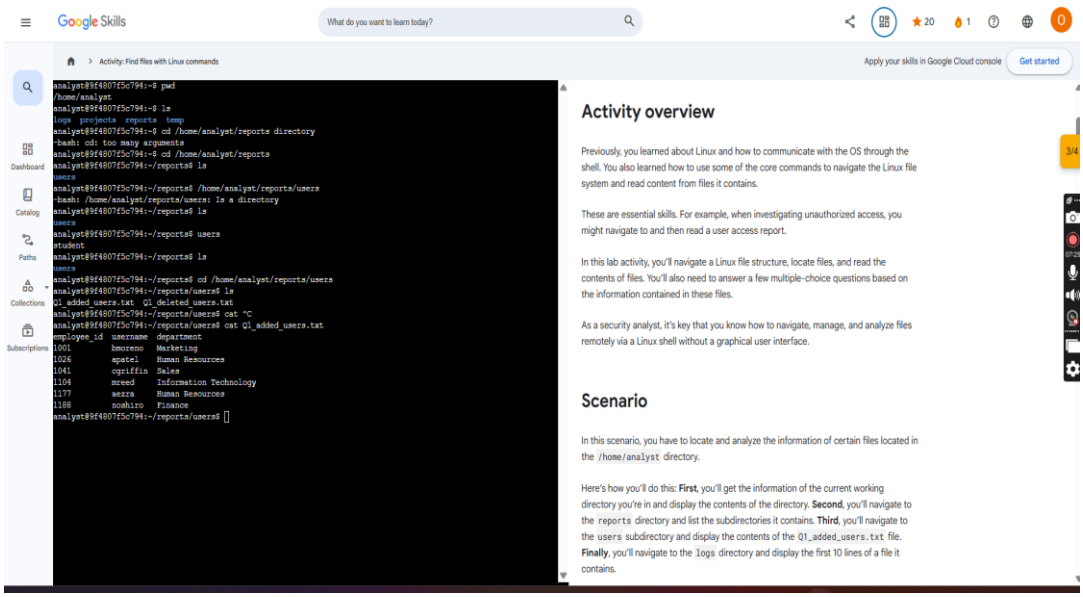
These are essential skills. For example, when investigating unauthorized access, you might navigate to and then read a user access report.

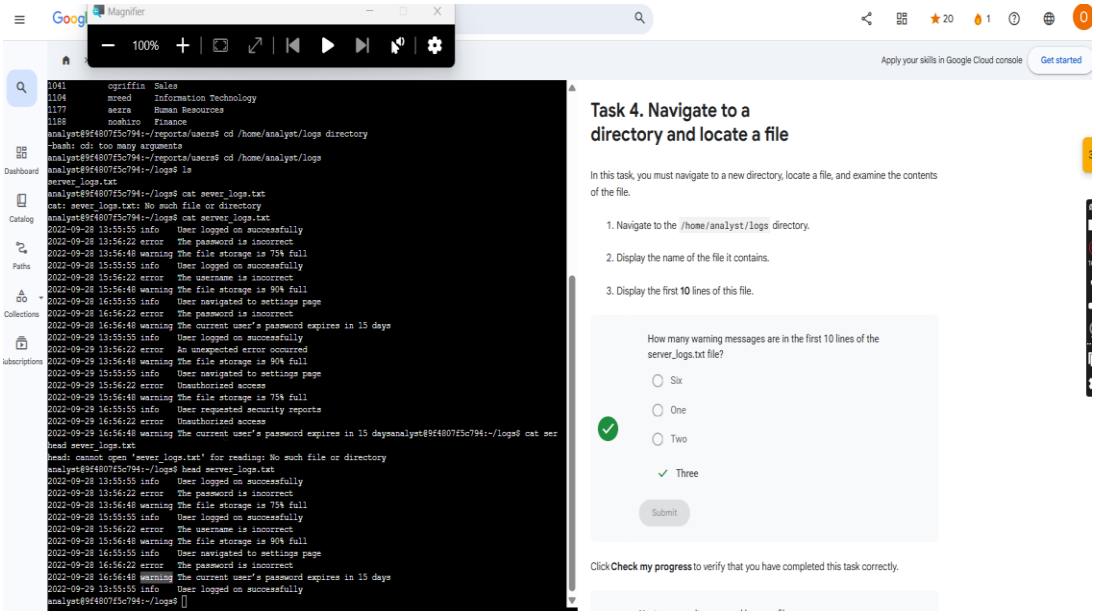
In this lab activity, you'll navigate a Linux file structure, locate files, and read the contents of files. You'll also need to answer a few multiple-choice questions based on the information contained in these files.

As a security analyst, it's key that you know how to navigate, manage, and analyze files remotely via a Linux shell without a graphical user interface.

This exemplar is a walkthrough of the previous Qwiklab activity, including detailed instructions and solutions. You may use this exemplar if you were unable to complete the lab and/or you need extra guidance in competing lab tasks. You may also refer to this exemplar to prepare for the graded quiz in this module.

Practice screenshot





Filter content in Linux

Filtering is selecting data that match a certain condition.

The **grep** command searches a specified file and returns all lines in the file containing a specified string or text. The **grep** command commonly takes two arguments: a specific string to search for and a specific file to search through.

example: **grep error time_logs.txt**

The pipe command is accessed using the pipe character (`|`). **Piping** sends the standard output of one command as standard input to another command for further processing.

For example, `ls /home/analyst/reports | grep users` returns the file and directory names in the `reports` directory that contain `users`. Before the pipe, `ls` indicates to list the names of the files and directories in `reports`. Then, it sends this output to the command after the pipe.

You can think of piping as a general tool that you can use whenever you want the output of one command to become the input of another command.

find

The `find` command searches for directories and files that meet specified criteria.

When using `find`, the first argument after `find` indicates where to start searching. For example, entering `find /home/analyst/projects` searches for everything starting at the `projects` directory.

Specifying criteria involves options. **Options** modify the behavior of a command and commonly begin with a hyphen (`-`).

-name and -iname

One key criteria analysts might use with `find` is to find file or directory names that contain a specific string. The specific string you're searching for must be entered in quotes after the `-name` or `-iname` options. The difference between these two options is that `-name` is case-sensitive, and `-iname` is not.

Note: An asterisk (`*`) is used as a wildcard to represent zero or more unknown characters.

-mtime

Security analysts might also use `find` to find files or directories last modified within a certain time frame. The `-mtime` option can be used for this search. For example, entering `find /home/analyst/projects -mtime -3` returns all files and directories in the `projects` directory that have been modified within the past three days.

The `-mtime` option search is based on days, so entering `-mtime +1` indicates all files or directories last modified more than one day ago, and entering `-mtime -1` indicates all files or directories last modified less than one day ago.

Note: The option `-mmin` can be used instead of `-mtime` if you want to base the search on minutes rather than days.

Lab Practice

Scenario

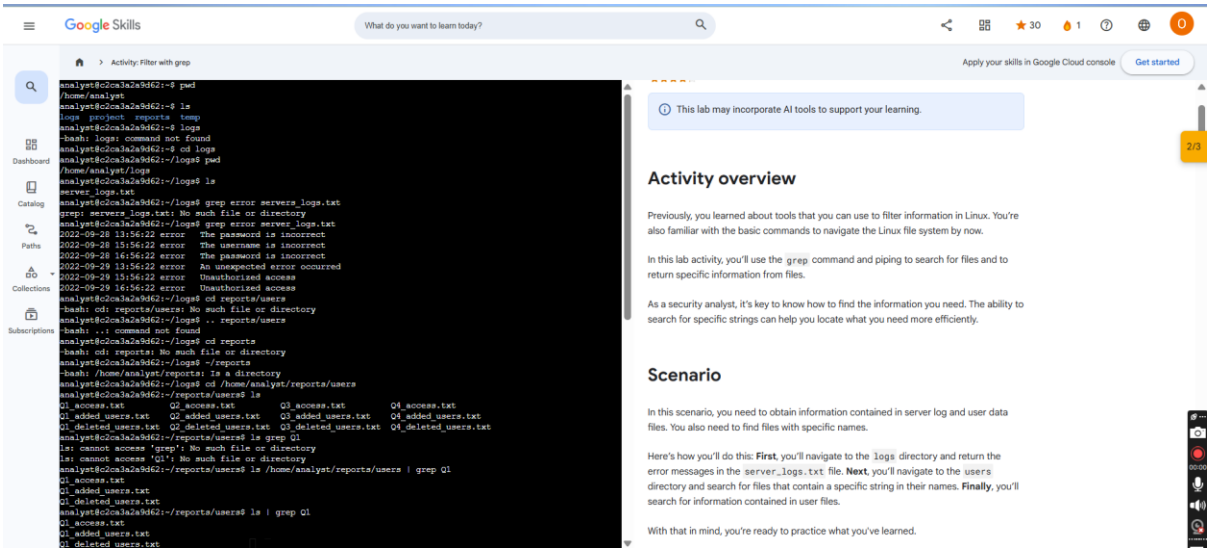
In this scenario, you need to obtain information contained in server log and user data files. You also need to find files with specific names.

Here's how you'll do this: First, you'll navigate to the logs directory and return the error messages in the server_logs.txt file. Next, you'll navigate to the users directory and search for files that contain a specific string in their names. Finally, you'll search for information contained in user files.

Solution: https://screenrec.com/share/J52VAg0ySB

commands used:

```
Task 1: Navigate to logs and find error messages

cd /home/analyst/logs

grep error server_logs.txt


# Task 2: Navigate to users reports and find files by name

cd /home/analyst/reports/users

ls | grep Q1

ls | grep access


# Task 3: Search inside user files

ls

grep jhill Q2_deleted_users.txt

grep "Human Resources" Q4_added_users.txt
```



Create and modify directories and files

`Mv – file name into (specify the directory you want to move it to starting from the root)`

`rmdir – remove directory`

The **`rmdir`** command cannot delete directories with files or subdirectories inside. For example, entering **`rmdir /home/analyst`** returns an error message.

`mkdir – create a directory(name of directory)`
`touch – creat a file(name and text format)`
`rm – file remove`
`cp – copy file to a different directory (does not delete the original)`

`nano – file editor. Nano (name of the file you want to edit)`
`you have to be in the file you want to edit.`
`Ctrl O – to save, enter to save with current user name. No auto-saving.`
`Ctrl x – exit`

You can also create a new file in nano by entering `nano` followed by a new file name. For example, entering `nano authorized_users.txt` from the `/home/analyst/reports` directory creates the `authorized_users.txt` file within that directory and opens it in a new nano editing window.
Note: Vim and Emacs are also popular command-line text editors.
`la`

In addition to the pipe (`|`), you can also use the right angle bracket (`>`) and double right angle bracket (`>>`) operators to redirect standard output.

When used with `echo`, the `>` and `>>` operators can be used to send the output of `echo` to a specified file rather than the screen. The difference between the two is that `>` overwrites your existing file, and `>>` adds your content to the end of the existing file instead of overwriting it. The `>` operator should be used carefully, because it's not easy to recover overwritten files.

When you're inside the directory containing the `permissions.txt` file, entering `echo "last updated date" >> permissions.txt` adds the string "last updated date" to the file contents. Entering `echo "time" > permissions.txt` after this command overwrites the entire file contents of `permissions.txt` with the string "time".

Note: Both the `>` and `>>` operators will create a new file if one doesn't already exist with your specified name.

Lab Activity: Manage files with Linux commands:

Video:

cd /home/analyst

mkdir logs

ls

rmdir temp

ls

cd notes

mv Q3patches.txt /home/analyst/reports/

ls /home/analyst/reports

rm tempnotes.txt

ls

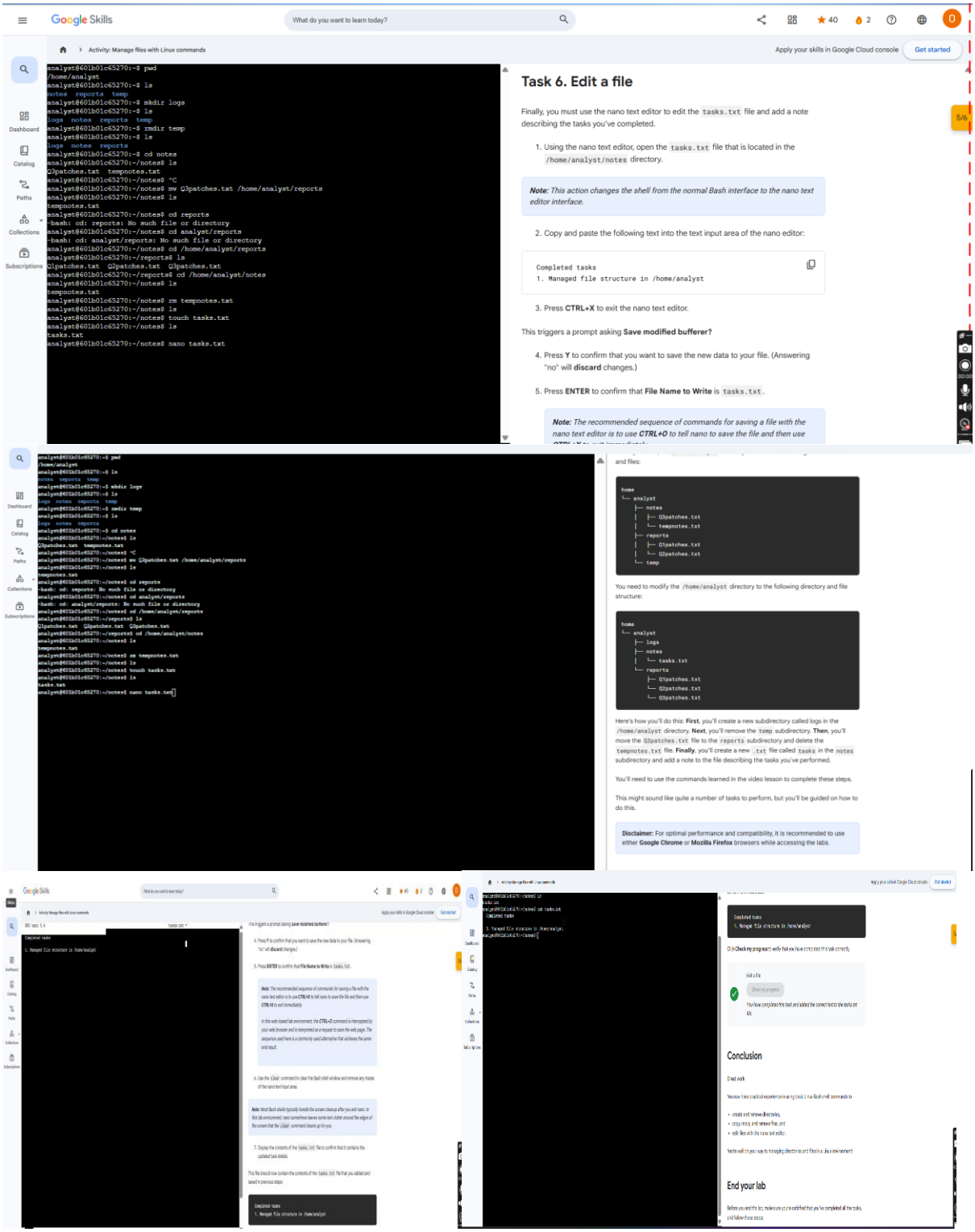
touch tasks.txt

ls

nano tasks.txt

clear

cat tasks.txt



File permissions and ownership

Ls – list

Ls -a = list hiddend files too

Ls -la = permissions to files and directory, including hidden files

- Ls -l = extended info on the file, permission and format. Displays permissions to files and directories. Also displays other additional information, including owner name, group, file size, and the time of last modification.
-

Change permissions: for The principle of least privilege

chmod = change mode

symbolic: chmod g+w, o-r access.txt

persmissions to who

g group

o other

u user

permition types

w write

r read

x execute

following command would add all permissions to `login_sessions.txt`:

```
chmod u+rwx,g+rwx,o+rwx login_sessions.txt
```

If you wanted to take all the permissions away, you could use

```
chmod u-rwx,g-rwx,o-rwx login_sessions.txt
```

Another way to assign these permissions is to use the equals sign (`=`) in this first argument. Using `=` with `chmod` sets, or assigns, the permissions exactly as specified. For example, the following command would set read permissions for `login_sessions.txt` for user, group, and other:

```
chmod u=r,g=r,o=r login_sessions.txt
```

Character	Description
`u`	indicates changes will be made to user permissions
`g`	indicates changes will be made to group permissions
`o`	indicates changes will be made to other permissions
`+`	adds permissions to the user, group, or other
`-`	removes permissions from the user, group, or other
`=`	assigns permissions for the user, group, or other

Note: When there are permission changes to more than one owner type, commas are needed to separate changes for each owner type. You should not add spaces after those commas.

The principle of least privilege in action

As a security analyst, you may encounter a situation like this one: There's a file called `bonuses.txt` within a compensation directory. The owner of this file is a member of the Human Resources department with a username of `hrrep1`. It has been decided that `hrrep1` needs access to this file. But, since this file contains confidential information, no one else in the `hr` group needs access.

You run `ls -l` to check the permissions of files in the compensation directory and discover that the permissions for `bonuses.txt` are `-rw-rw----`. The group owner type has read and write permissions that do not align with the principle of least privilege.

To remedy the situation, you input `chmod g-rw bonuses.txt`. Now, only the user who needs to access this file to carry out their job responsibilities can access this file.

Lab practice: doc link: [Google Doc Link]

Code:

cd /home/researcher2/projects

ls -l

ls -la

chmod o-w project_k.txt

ls -l

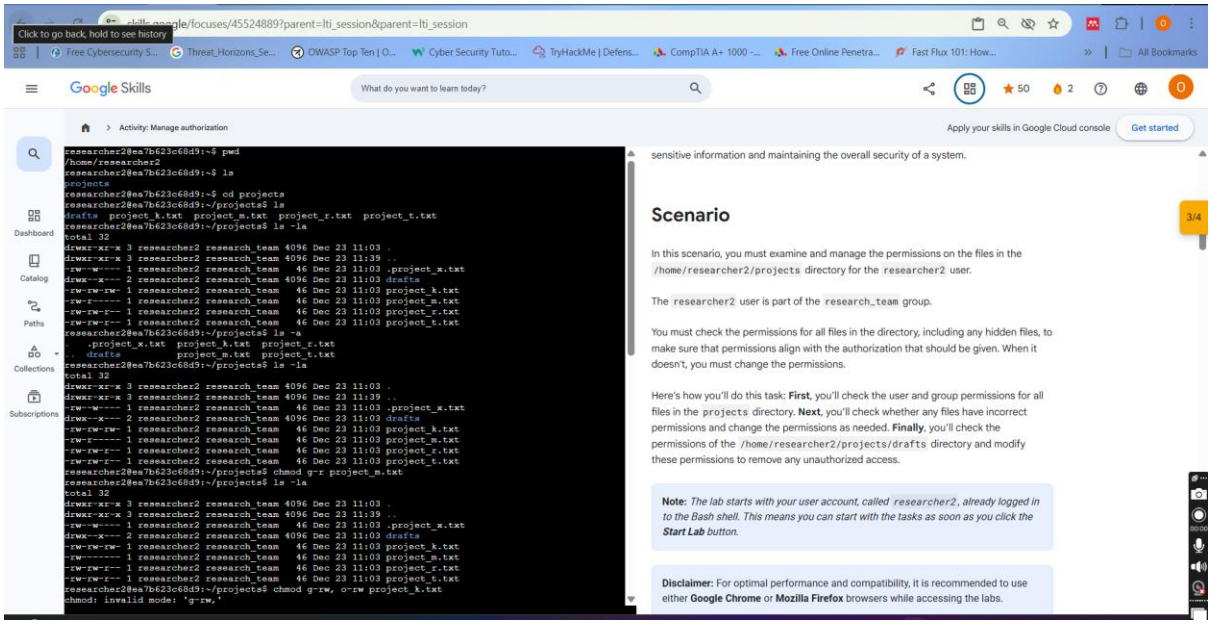
chmod g-r project_m.txt

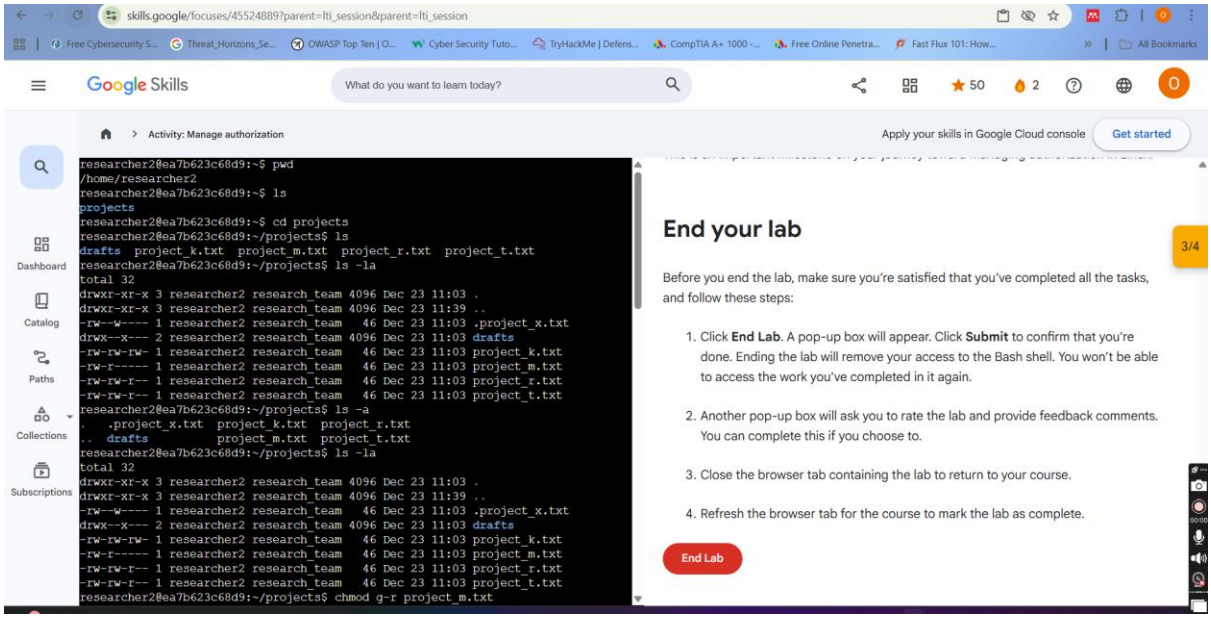
ls -la

chmod u-w,g-w,g+r .project_x.txt

ls -l

chmod g-x drafts





Add and delete users

Types of users

Root user – supper user. sudo provides control to root user access. Special privileges.

Useradd = sudo useradd osita

Userdel – sudo userdel osita

- `-g`: Sets the user's default group, also called their primary group
- `-G`: Adds the user to additional groups, also called supplemental or secondary groups

To use the `-g` option, the primary group must be specified after `-g`. For example, entering `sudo useradd -g security fgarcia` adds `fgarcia` as a new user and assigns their primary group to be `security`.

To use the `-G` option, the supplemental group must be passed into the command after `-G`. You can add more than one supplemental group at a time with the `-G` option. Entering `sudo useradd -G finance,admin fgarcia` adds `fgarcia` as a new user and adds them to the existing `finance` and `admin` groups.

`usermod`:

usermod

The `usermod` command modifies existing user accounts. The same `-g` and `-G` options from the `useradd` command can be used with `usermod` if a user already exists.

To change the primary group of an existing user, you need the `-g` option. For example, entering `sudo usermod -g executive fgarcia` would change `fgarcia`'s primary group to the `executive` group.

To add a supplemental group for an existing user, you need the `-G` option. You also need a `-a` option, which appends the user to an existing group and is only used with the `-G` option. For example, entering `sudo usermod -a -G marketing fgarcia` would add the existing `fgarcia` user to the supplemental `marketing` group.

Note: When changing the supplemental group of an existing user, if you don't include the `-a` option, `-G` will replace any existing supplemental groups with the groups specified after `usermod`. Using `-a` with `-G` ensures that the new groups are added but existing groups are not replaced.

`Chown`

There are other options you can use with `usermod` to specify how you want to modify the user, including:

- `-d`: Changes the user's home directory.
- `-l`: Changes the user's login name.
- `-L`: Locks the account so the user can't log in.

The `userdel` command deletes a user from the system.

The `chown` command changes ownership of a file or directory. You can use `chown` to change user or group ownership. To change the user owner of the `access.txt` file to `fgarcia`, enter `sudo chown fgarcia access.txt`. To change the group owner of `access.txt` to `security`, enter `sudo chown :security access.txt`. You must enter a colon (`:`) before `security` to designate it as a group name.

Task

sudo useradd researcher9

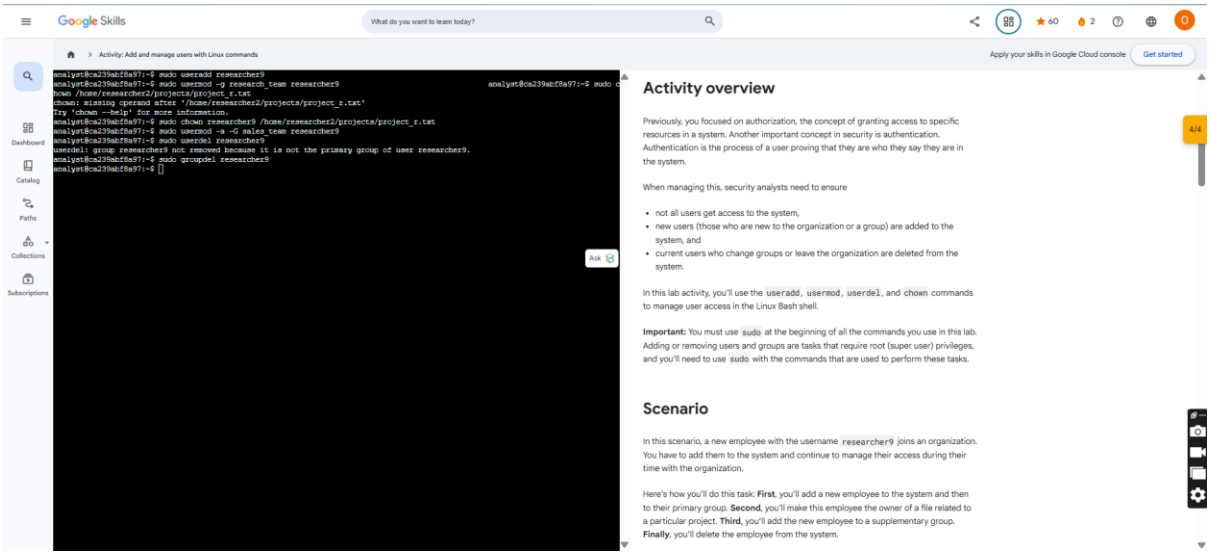
sudo usermod -g research_team researcher9

sudo chown researcher9 /home/researcher2/projects/project_r.txt

sudo usermod -a -G sales_team researcher9

sudo userdel researcher9

sudo groupdel researcher9



Resources:

man = manual (man usermod)

whatis = ehat a command does (eg: whatis tail_

apropos = -**a** or with key word for specificity. when you don't know what to look up…. Eg: how to change password, use the word "password"

output of the **`man`** command is also called a "man page."

Practical Application in Lab

codes

whatis cat

man cat

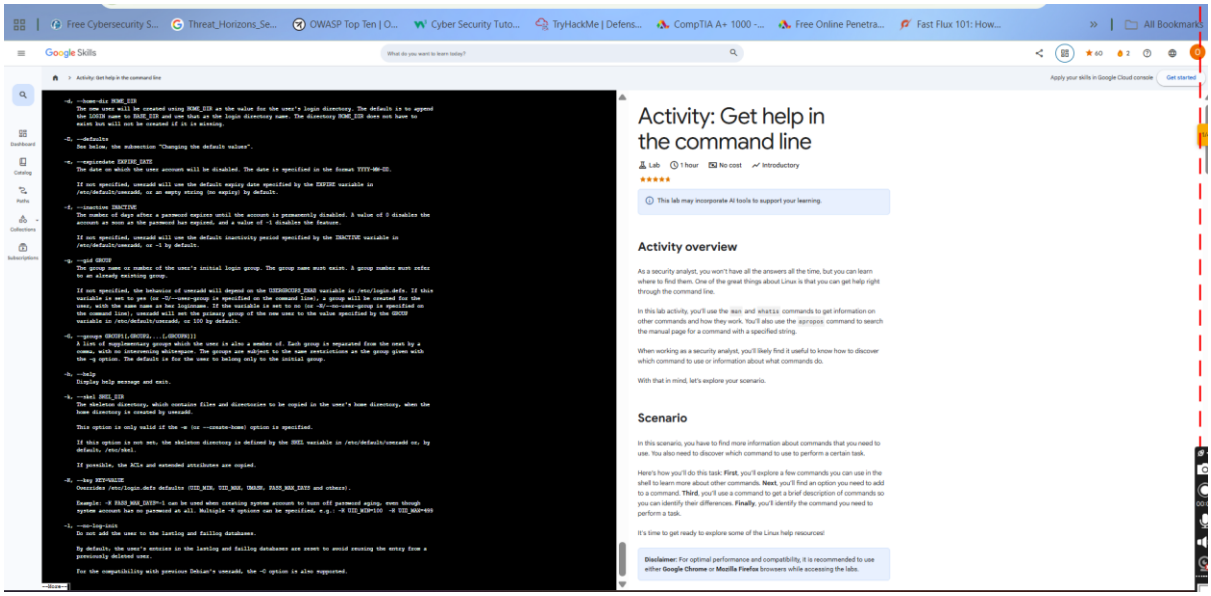
apropos -a first part file

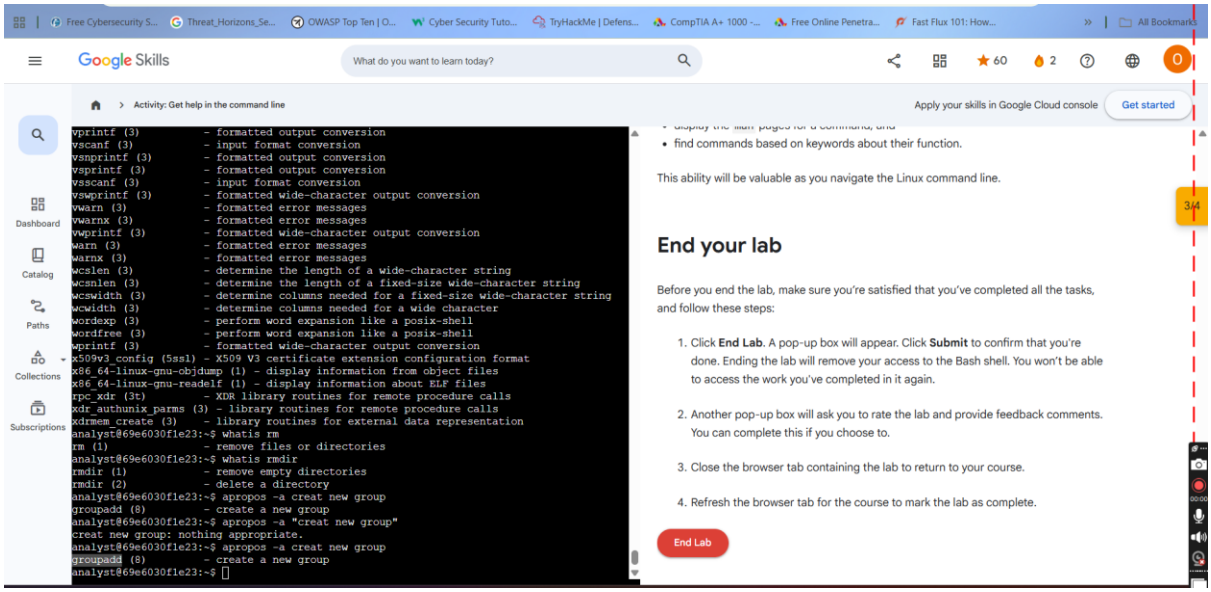
man useradd

whatis rm

whatis rmdir

apropos -a create new group





Terms and definitions from Course 4, Module 3

Absolute file path: The full file path, which starts from the root

Argument (Linux): Specific information needed by a command

Authentication: The process of verifying who someone is

Authorization: The concept of granting access to specific resources in a system

Bash: The default shell in most Linux distributions

Command: An instruction telling the computer to do something

File path: The location of a file or directory

Filesystem Hierarchy Standard (FHS): The component of the Linux OS that organizes data

Filtering: Selecting data that match a certain condition

nano: A command-line file editor that is available by default in many Linux distributions

Options: Input that modifies the behavior of a command

Permissions: The type of access granted for a file or directory

Principle of least privilege: The concept of granting only the minimal access and authorization required to complete a task or function

Relative file path: A file path that starts from the user's current directory

Root directory: The highest-level directory in Linux

Root user (or superuser): A user with elevated privileges to modify the system

Standard input: Information received by the OS via the command line

Standard output: Information returned by the OS through the shell