Lab 1: Access Control Lab

Mitch Kuebel

Nur Syahirah Mohamad Sabri

Question 1: Creating user

User Creation Method:

The generalized method that is used to create the users is by using `adduser` command. The basic syntax for the command is as follows: `sudo adduser [username]` (GeeksforGeeks, 2023). The `sudo` command allows to run the command with administrator privileges while `adduser` is used to add a user.

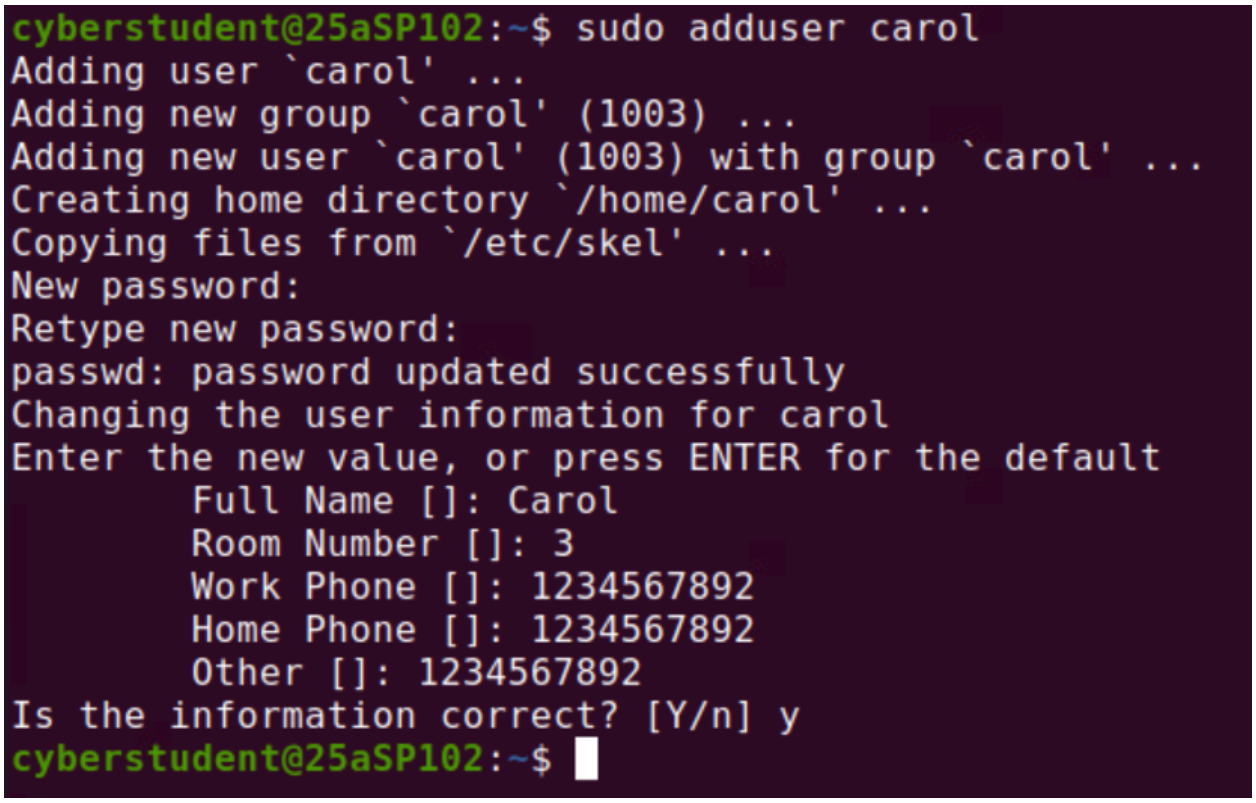

Figure 1: Creating user carol

Then, a password was created for carol. Then, the terminal requested to check the information for the user added was correct and entered “Y” to confirm the information was correct and the process of adding a user was complete. (*How to Add and Remove Users on Ubuntu 20.04*, 2020)

The command and the process above was repeated to add all the users which are Bob, Alice, Eve, Malory, Trent, Walter, Victor and Hands.

User Creation Evidence:

The method to verify these users' existence is by using `ls -lh` command, it will display all the users that have just been created. (in, 2012).

```
cyberstudent@25aSP102:/home$ ls -lh
total 40K
drwxr-xr-x  5 alice        alice        4.0K Feb  7 12:47 alice
drwxr-xr-x  2 bob          bob          4.0K Jan 24 12:04 bob
drwxr-xr-x  6 carol        carol        4.0K Feb  7 12:46 carol
drwxr-xr-x 21 cyberstudent cyberstudent 4.0K Feb 14 12:39 cyberstudent
drwxr-xr-x  3 eve          eve          4.0K Jan 31 12:46 eve
drwxr-xr-x  2 hands        hands        4.0K Jan 24 12:16 hands
drwxr-xr-x  3 malory       malory       4.0K Jan 31 12:52 malory
drwxr-xr-x  2 trent        trent        4.0K Jan 24 12:12 trent
drwxr-xr-x  2 victor       victor       4.0K Jan 24 12:15 victor
drwxr-xr-x  2 walter       walter       4.0K Jan 24 12:14 walter
cyberstudent@25aSP102:/home$
```

Figure 2: Shows a screenshot of all the users that created in directory

User Creation Audit:

In order to test that the users were successfully created, the `su` command was used to attempt to change into each user (in, 2012) as shown in Figure 3 below

```
cyberstudent@25aSP102:~$ su - alice
Password:
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

alice@25aSP102:~$
```

Figure 3: Changing to user “alice”

User Creation Evidence:

We made sure that the users existed by checking the /etc/passwd file `cat /etc/passwd` will output contents of /etc/passwd to the console. This file stores information on all users on a linux machine. The output of this shows the users have

been created. (GeeksforGeeks, 2024). The command `tail cat` in Figure 4 below was used to display the last few lines of the content in file /etc/passwd which will display all the users that were added.

```
cyberstudent@25aSP102:~$ tail cat /etc/passwd
tail: cannot open 'cat' for reading: No such file or directory
==> /etc/passwd <==
fwupd-refresh:x:129:136:fwupd-refresh user,,,:/run/systemd:/usr/sbin/nologin
alice:x:1001:1001:Alice,1,1234567890,1234567890,1234567890:/home/alice:/bin/bash
bob:x:1002:1002:Bob,2,1234567891,1234567891,1234567891:/home/bob:/bin/bash
carol:x:1003:1003:Carol,3,1234567892,1234567892,1234567892:/home/carol:/bin/bash
eve:x:1004:1004:Eve,4,1234567893,1234567893,1234567893:/home/eve:/bin/bash
malory:x:1005:1005:Malory,5,1234567894,1234567894,1234567894:/home/malory:/bin/bash
trent:x:1006:1006:Trent,6,1234567895,1234567895,1234567895:/home/trent:/bin/bash
walter:x:1007:1007:Walter,7,1234567896,1234567896,1234567896:/home/walter:/bin/bash
victor:x:1008:1008:Victor,8,1234567897,1234567897,1234567897:/home/victor:/bin/bash
hands:x:1009:1009:Hands,9,1234567898,1234567898,1234567898:/home/hands:/bin/bash
cyberstudent@25aSP102:~$
```

Figure 4: tail cat command was used to display the added users

Grant Admin Method:

`sudo adduser [username] sudo` - Adds user, [username], to the sudo group, which allows administrator / root privileges through the sudo command. (GeeksforGeeks, 2023). Refer to Figure 5 below for a screenshot of adding a user to the sudo group.

```
cyberstudent@25aSP102:~$ sudo adduser hands sudo
Adding user `hands' to group `sudo' ...
Adding user hands to group sudo
Done.
cyberstudent@25aSP102:~$
```

Figure 5: Shows a screenshot of adding user to sudo group

This command was repeated for Alice, and Bob as they needed admin access too. Malory and Trent, both do not have admin access. Although Malory works in accounting and must handle financial transactions and records, she does not need administrator access, which would typically grant her extensive control over the system. Giving her administrator access may put the company's sensitive settings or data at danger of accidental or malicious alteration. On the other hand, Trent, who works in R&D, also needs access to resources pertaining to research, but not system-wide settings. Granting him administrative access could jeopardize data integrity by causing unforeseen changes in other crucial business areas like operations or finance. If these access levels are miscalculated, the business may be vulnerable to data breaches,

illegal modifications, or system malfunctions that could seriously affect its operational and financial viability.

Grant Admin Evidence:

`getent group sudo` will output the information about each user that specifically is in the sudo group, which allows root / administrator access through the sudo command. [getent] is used to “get entries” from databases, [group] the database to read, [sudo] the entry to look for (GeeksforGeeks, 2024). From figure 6 below, it shows that the administrative privilege has been set to Alice, Bob and Hands

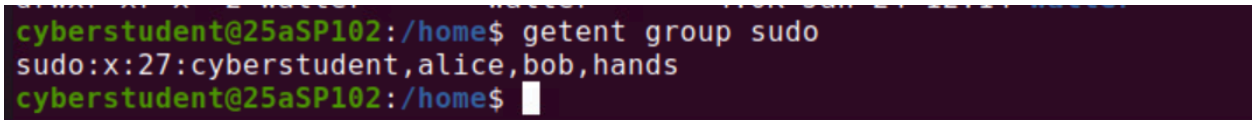

Figure 6: shows screenshot of all the users that have admin access

Password Forced Method:

Since the users have the temporary password when creating the users, the system will require them to change their passwords when they first log in to the system. The `chage` command which means “change age” is used to manage account aging parameters and require users to update their passwords periodically. (Sethusubramanian, 2022). In Figure 7 below, the `chage -d 0 [username]` command is used to force users to change their password immediately after logging in for the first time while `chage -l` lists the password aging details.

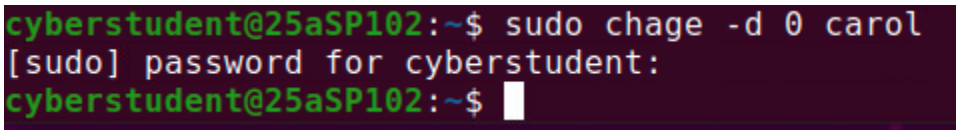

Figure 7: shows screenshot of force user to change password

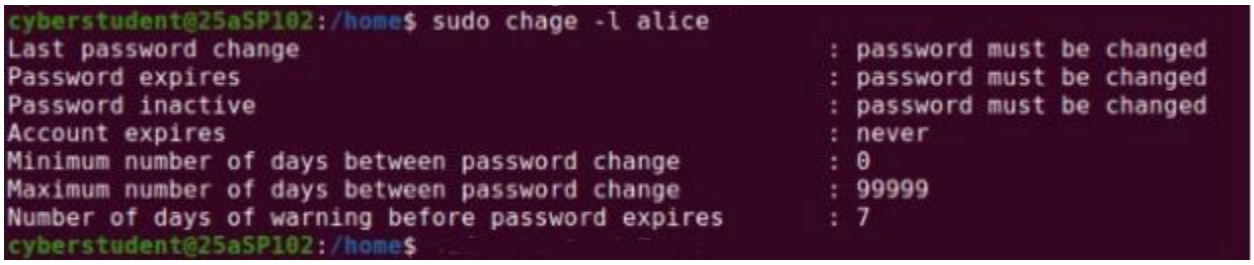

Figure 8: shows screenshot of password details and aging parameter

Password Forced Evidence:

To check if the password policy is working, we login to one of the user's accounts by using command in Figure 9 below. The command `su` switch the user where it allows to switch from current user to another user.

Figure 9: shows screenshot of user Malory need to change the password

Once running the command, the system asked for the temporary password that was set and then let the user know that the user is required to change their password immediately. Then, the system asks for the temporary password again, then asks for the new password and the confirmation of the new password. This will change the password successfully. (GeeksforGeeks, 2023).

Question 2: Permission Bits Review

Qualitative Regular Permissions:

In linux, permissions control how users and groups can access files and directories. The system uses a qualitative permission model where different types of permissions are applied to users, groups and others. There are three basic types of permissions which are read (r), write(w) and execute (x) (NERSC, n.d.). Figure 10 shows how a 3-bit binary number corresponds to an octal number and the permissions that the combination represents.

Binary Bit Permutation	Octal Bits	Qualitative Description
000	0	No Permissions allowed, representing (- - -) that users can not read, write and execute the file (NERSC, n.d.)
001	1	Execute permission only, representing (- - x) users can run the file but can not view or modify its content (NERSC, n.d.)
010	2	Write permission only, representing (- w -) users can modify a file but cannot view its content (NERSC, n.d.)
011	3	Write and Execute permission only, representing (- w x) users can run and modify the file but can not view its content (NERSC, n.d.)
100	4	Read permission only, representing (r - -) users can view but not modify or execute the file (NERSC, n.d.)
101	5	Read and Execute only, representing (r - x) users can view and run the file but can not modify its content (NERSC, n.d.)
110	6	Read and Write only, representing (r w -) users can view and modify the file but can not execute it (NERSC, n.d.)
111	7	Read, Write, and Execute, which also means all permissions allowed, representing (r w x) users can view, modify and execute the file (NERSC, n.d.)

Figure 10: Permutation of three binary bits

Qualitative Special Permissions:

In linux, special permissions add an extra layer of control beyond the regular read, write and execute permissions. These special permissions are Set User ID (SUID), Set Group ID (SGID) and the sticky bit. (Blinkii, 2020). Figure 11 shows how a 3-bit binary number corresponds to an octal number and the special permission that octal bits represent.

Binary Bit Permutation	Octal Bits	Qualitative Description
000	0	No any special permission allowed (Blinkii, 2020)
001	1	Sticky bit is used when subdirectories and files under a directory that has the sticky bit set can only be removed by the root user, the directory owner, or the file owner. In a shared folder where everyone has read, write, and execute access, this unique permission helps keep users from erasing other users' files. (Blinkii, 2020)
010	2	SGID allows the file execution with the privileges of the group which owns the file. All newly generated files or subdirectories beneath the directory will inherit the same group ownership as the directory itself if the SGID bit is set (Blinkii, 2020)
011	3	SGID and Sticky allow the file's execution with the group's privileges and restrict the file deletion (Blinkii, 2020)
100	4	SUID allows the file execution with the user's privileges (Blinkii, 2020)
101	5	SUID and Sticky allow the file execution with the user's privileges and restrict the file deletion (Blinkii, 2020)
110	6	SUID and SGID allow the file execution with both the user's and group's privileges. The files retain the group ownership while SUID has no effect (Blinkii, 2020)
111	7	Sticky, SGID, SUID allow the execution of files with both the user's and group's privileges and restrict the file deletion to the file owners (Blinkii, 2020)

Figure 11: Permutation of special bit permissions

Question 3: Creating Groups

Group Creation Method:

To create the company's user group, the `sudo groupadd [groupName]` command was used. `sudo` executes the following command with root privileges, `groupadd` command for creating new groups, `groupName` is the name of the group to be created. (GeeksforGeeks, 2023). This can be shown from the figure 12 below.

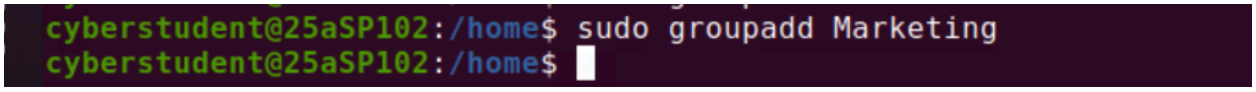

Figure 12: shows screenshot of creating the users group

This command was repeated to create the following groups: Security, Sales, receptionist, Accounting, RandD, CustomerService, and CEO.

To show the list of groups created was by using `cut -d: -f1 /etc/group | tail` command. The `cut` extract sections from each line of a file or output, `-d:` specifies delimiter that separates fields in each line, `-f1` tells cut command which field to extract from. In this case, -f1 specifies the first field of /etc/field, which contains groups. `/etc/group` = file that contains the group account information, `|` = pipe, takes output of left command and feeds input to right command and `tail` shows only the last few lines of the output. (Aleksic, 2019)

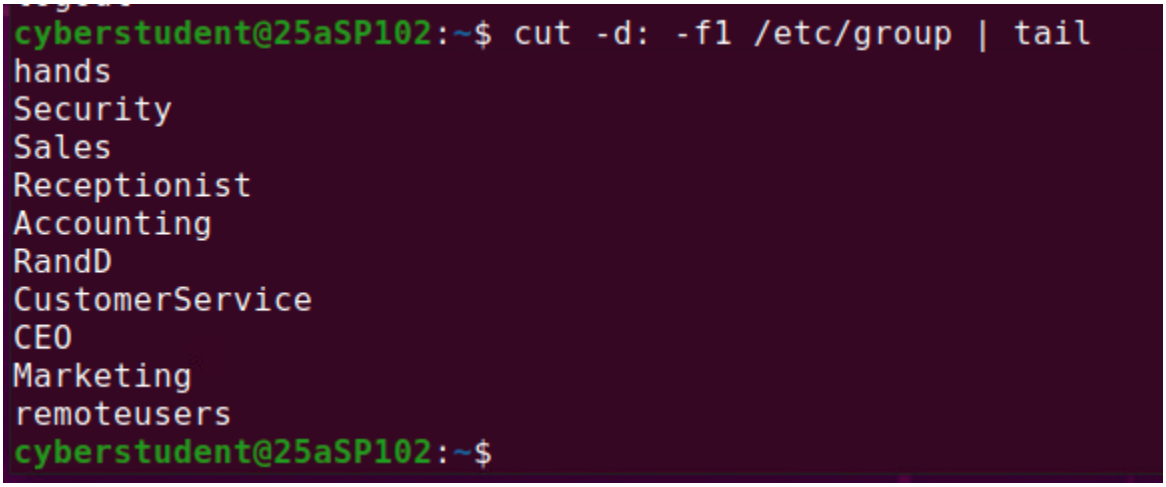

Figure 13: Evidence of all groups created

Group Membership Evidence:

To add users to a specific group, the command `sudo usermod -aG [groupName] [user]` was used. `sudo` executes the following command with root privileges, `usermod` command used to modify user attributes, `groupName` is a group for `user` to be added to. (GeeksforGeeks, 2023)

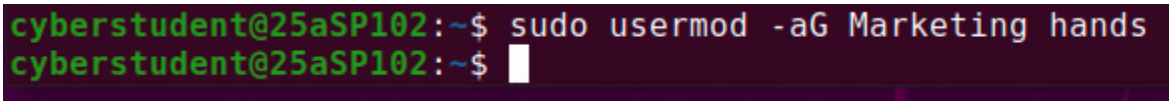

Figure 14: shows the command to add the users to the specific group

The process is repeated to add all the users into their appropriate group. The users and the groups they are added to is as following:

User	**Groups the user are added to**
Alice	Security
Bob	Security
Carol	Sales
Eve	Receptionist
Malory	Accounting
Trent	RandD
Walter	RandD
Victor	CustomerService
Hands	CEO, RandD, Marketing

Figure 15: Groups that users are added to

```
cyberstudent@25aSP102:~$ getent group Security
Security:x:1010:alice,bob
cyberstudent@25aSP102:~$
```

Figure 16: shows all users in the group Security

```
cyberstudent@25aSP102:~$ getent group Sales
Sales:x:1011:carol
cyberstudent@25aSP102:~$
```

Figure 17: shows all users in the group Sales

```
cyberstudent@25aSP102:~$ getent group Receptionist
Receptionist:x:1012:eve
cyberstudent@25aSP102:~$
```

Figure 18: shows all users in the group Receptionist

```
cyberstudent@25aSP102:~$ getent group Accounting
Accounting:x:1013:malory
cyberstudent@25aSP102:~$
```

Figure 19: shows all users in the group Accounting

```
cyberstudent@25aSP102:~$ getent group RandD
RandD:x:1015:trent,walter,hands
cyberstudent@25aSP102:~$
```

Figure 20: shows all users in the group RandD

```
cyberstudent@25aSP102:~$ getent group CustomerService
CustomerService:x:1016:victor
cyberstudent@25aSP102:~$
```

Figure 21: shows all users in the group CustomerService

```
cyberstudent@25aSP102:~$ getent group CEO
CEO:x:1017:hands
cyberstudent@25aSP102:~$
```

Figure 22: shows all users in the group CEO

```
cyberstudent@25aSP102:~$ getent group Marketing
Marketing:x:1018:hands
cyberstudent@25aSP102:~$
```

Figure 23: shows all users in the group Marketing

Directory Structure Reasoning:

A directory is a container that is used to contain folders and files. It organizes files and folders in a hierarchical manner. In other words, directories are like folders that help organize files on a computer. Just like you use folders to keep your papers and documents in order, the operating system uses directories to keep track of files and where they are stored. Different structures of directories can be used to organize these files, making it easier to find and manage them. (*Structures of Directory in Operating System - GeeksforGeeks*, 2018). The directory was structured to fit the requirements stated in the manual. File names contain no spaces and are all in lowercase letters. All files are also only owned by root and not a specific user. The number of folders in the groupshare directory also only increased by 1.

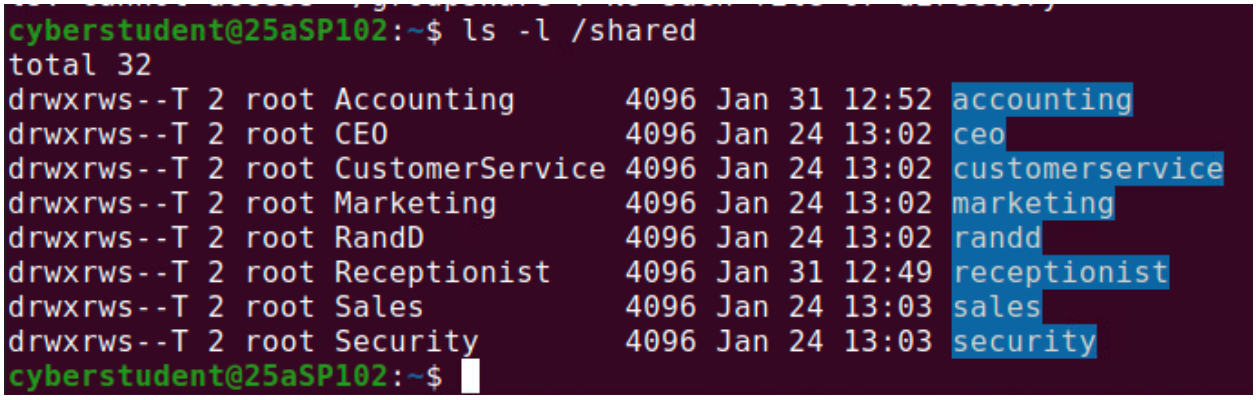

Figure 24: shows ownership of all folders in “shared” directory

Directory Creation Method:

Firstly, the main directory was created and named as shared. The commands used: `sudo mkdir [directory path]`. `sudo` runs the command with admin privilege, `mkdir` stands for make directory and is used to create a new directory, `[directory path]` is the path where the directory will be created.

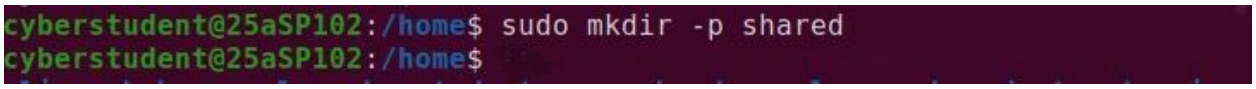

Figure 25: shows screenshot of creating a new folder for group shares

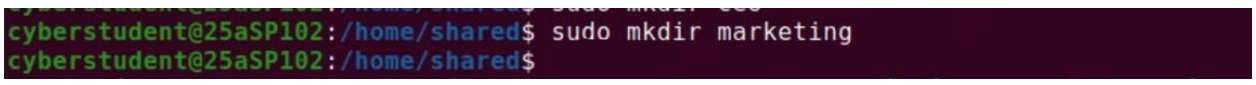

Figure 26: shows screenshot of creating subdirectories for specific group

Therefore, the Marketing directory will be created directly under the /shared directory. This command was used to create files for all the groups which are sales, receptionist, accounting, security, randd (Research & Development), customerservice, ceo. (Abhishek, 2022)

```
cyberstudent@25aSP102:/home/shared$ ls
accounting  ceo  customerservice  marketing  randd  receptionist  sales  security
cyberstudent@25aSP102:/home/shared$
```

Figure 27: shows screenshot of user groups has been created in shared directory

Evidence of File Structure:

The figure below shows all the file directories and the required files that was created to to have an appropriate file structure.

```
cyberstudent@25aSP102:~$ sudo tree /shared
/shared
├── accounting
│   └── accounting.txt
├── ceo
│   └── ceo.txt
├── customerservice
│   └── customerservice.txt
├── marketing
│   └── marketing.txt
├── randd
│   └── randd.txt
├── receptionist
│   └── receptionist.txt
├── sales
│   └── sales.txt
└── security
    └── security.txt

8 directories, 8 files
cyberstudent@25aSP102:~$
```

Figure 28: shows the directory structure by using `tree` command

Sample File Creation:

Sample file was created in the file directories. First, the directory was changed using the command `cd` to the intended directory that the file wanted to be created. The command `touch` was used to create a text file (Abishek, 2022).

```
cyberstudent@25aSP102:/home/shared/ceo$ sudo touch ceo.txt
cyberstudent@25aSP102:/home/shared/ceo$
```

Figure 29: Making a test file

This step is repeated to create sample files in all the file directories in /shared directory. The list of the sample file created can be viewed in Figure 30 below.

```
cyberstudent@25aSP102:/home/shared/accounting$ cd ../
cyberstudent@25aSP102:/home/shared$ cd ceo
cyberstudent@25aSP102:/home/shared/ceo$ sudo touch ceo.txt
cyberstudent@25aSP102:/home/shared/ceo$ cd ../customerservice/
cyberstudent@25aSP102:/home/shared/customerservice$ sudo touch customerservice.txt
cyberstudent@25aSP102:/home/shared/customerservice$ cd ../marketing/
cyberstudent@25aSP102:/home/shared/marketing$ sudo touch marketing.txt
cyberstudent@25aSP102:/home/shared/marketing$ cd ../randd/
cyberstudent@25aSP102:/home/shared/randd$ sudo touch randd.txt
cyberstudent@25aSP102:/home/shared/randd$ cd ../receptionist/
cyberstudent@25aSP102:/home/shared/receptionist$ sudo touch receptionist.txt
cyberstudent@25aSP102:/home/shared/receptionist$ cd ../sales/
cyberstudent@25aSP102:/home/shared/sales$ sudo touch sales.txt
cyberstudent@25aSP102:/home/shared/sales$ cd ../security/
cyberstudent@25aSP102:/home/shared/security$ sudo touch security.txt
cyberstudent@25aSP102:/home/shared/security$ ls
security.txt
cyberstudent@25aSP102:/home/shared/security$ ls ../marketing/
marketing.txt
cyberstudent@25aSP102:/home/shared/security$ ls ../receptionist/
receptionist.txt
cyberstudent@25aSP102:/home/shared/security$ ls ../customerservice/
customerservice.txt
cyberstudent@25aSP102:/home/shared/security$ ls ../sales
sales.txt
cyberstudent@25aSP102:/home/shared/security$ ls ../ceo
ceo.txt
cyberstudent@25aSP102:/home/shared/security$ ls ../accounting/
accounting.txt
cyberstudent@25aSP102:/home/shared/security$ ls ../randd
randd.txt
cyberstudent@25aSP102:/home/shared/security$
```

Figure 30: shows screenshot of creating text file in each directory

Explanation of Permission:

We chose to set the permissions for each department share as 2770 (in octal). The octal permissions are: number 2 means that Set GID, which ensures that all new files inside this directory are automatically owned by the specific group. Number 7 means that the owner has the full access that includes read, write, and execute files. The next number 7 means that the group has full access that also includes read, write and execute files. Last, number 0 means that the others have no permission to read, write and execute the file.

The directories in the /shared directory were given the 1770 permission (in octal). The octal permissions: 1 is a sticky bit that ensures that only the file owner can delete the files, 7 which means that the owner has full permissions (read, write, execute). The next 7 is that the group has full permissions and 0 means that the others have no permission. (GeeksforGeeks, 2024)

Permission Method:

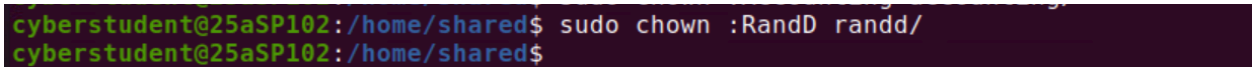

Figure 31: shows screenshot of setting group ownership on directories

From figure 30, this is a command that only group members can access the group share folder. For example, in randd, only randd group members can access it.

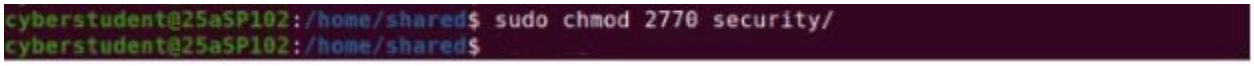

Figure 32: shows screenshot of setting the octal permission

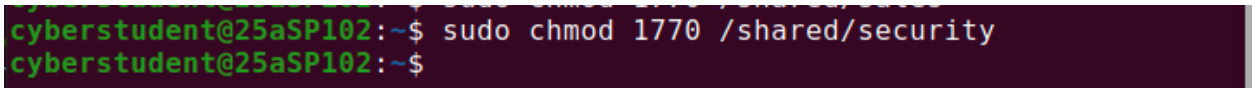

Figure 33: shows command to prevent users from deleting files they didn't create

Permission Evidence:

```
cyberstudent@25aSP102:~$ sudo ls -lh /shared/*
/shared/accounting:
total 0
-rw-r--r-- 1 root root 0 Jan 24 13:01 accounting.txt

/shared/ceo:
total 0
-rw-r--r-- 1 root root 0 Jan 24 13:02 ceo.txt

/shared/customerservice:
total 0
-rw-r--r-- 1 root root 0 Jan 24 13:02 customerservice.txt

/shared/marketing:
total 0
-rw-r--r-- 1 root root 0 Jan 24 13:02 marketing.txt

/shared/randd:
total 0
-rw-r--r-- 1 root root 0 Jan 24 13:02 randd.txt

/shared/receptionist:
total 0
-rw-r--r-- 1 root root 0 Jan 24 13:02 receptionist.txt

/shared/sales:
total 0
-rw-r--r-- 1 root root 0 Jan 24 13:03 sales.txt

/shared/security:
total 0
-rw-r--r-- 1 root root 0 Jan 24 13:03 security.txt
cyberstudent@25aSP102:~$
```

Figure 34: shows the permission access in the directory

Based on the Figure 35 below, it shows that each department now has its own shared folder and only users belonging to the department's group can access its folder. Non-group users, on the other hand, have no access (`---`) which prevents unauthorized access. The setgid bit (`s`) ensures that group ownership is retained for new files. (GeeksforGeeks, 2024)

```
cyberstudent@25aSP102:/home/shared$ ls -lh
total 32K
drwxrws--- 2 root Accounting      4.0K Jan 24 13:01 accounting
drwxrws--- 2 root CEO             4.0K Jan 24 13:02 ceo
drwxrws--- 2 root CustomerService 4.0K Jan 24 13:02 customerservice
drwxrws--- 2 root Marketing       4.0K Jan 24 13:02 marketing
drwxrws--- 2 root RandD           4.0K Jan 24 13:02 randd
drwxrws--- 2 root Receptionist    4.0K Jan 24 13:02 receptionist
drwxrws--- 2 root Sales           4.0K Jan 24 13:03 sales
drwxrws--- 2 root Security        4.0K Jan 24 13:03 security
cyberstudent@25aSP102:/home/shared$
```

Figure 35: shows screenshot of group ownership has been set

In order to test that file and directory permissions were up to spec, we decided to first log in as a user we had created in Figure 36 below. When Malory first logged in, she was required to change the temporary password as it was enforced in the previous task. Malory was able to create and modify a file named malory.txt, and even modify a file named accounting.txt, as she's in accounting group user. However, she could not be able to delete the accounting.txt file as she does not have permission to delete the file that wasn't hers. She also could not access the receptionist shared folder as she is not in that group. Last, she tried to open up a directory the user was not authorized for, which as seen below Malory did not have permission to change her directory to marketing.

```
cyberstudent@25aSP102:~$ su malory
Password:
You are required to change your password immediately (administrator enforced)
Changing password for malory.
Current password:
New password:
Retype new password:
malory@25aSP102:/home/cyberstudent$
```

Figure 36.1: switched the user to Malory

```
malory@25aSP102:/shared/accounting$ touch malory.txt
malory@25aSP102:/shared/accounting$
```

Figure 36.2: created Malory's owned file

```
malory@25aSP102:/shared/accounting$ nano malory.txt
malory@25aSP102:/shared/accounting$
```

Figure 36.3: edited Malory's owned file

```
malory@25aSP102:/shared/accounting$ cat malory.txt
hello, this was written by malory using nano
malory@25aSP102:/shared/accounting$
```

Figure 36.4: displayed the content of Malory's owned file

```
malory@25aSP102:/shared/accounting$ nano accounting.txt
malory@25aSP102:/shared/accounting$
```

Figure 36.5: edited group share file

```
malory@25aSP102:/shared/accounting$ rm -r ../accounting/
rm: remove write-protected regular empty file '../accounting/accounting.txt'?
rm: cannot remove '../accounting/': Permission denied
malory@25aSP102:/shared/accounting$
```

Figure 36.6: tried to remove not owned group share file

```
malory@25aSP102:/shared$ cd receptionist/
bash: cd: receptionist/: Permission denied
malory@25aSP102:/shared$
```

Figure 36.7: tried to access other group share folder

```
malory@25aSP102:/shared$ touch malory2.txt receptionist/
touch: cannot touch 'malory2.txt': Permission denied
touch: setting times of 'receptionist/': Permission denied
malory@25aSP102:/shared$
```

Figure 36.8: tried to create a file in unauthorized share folder

Question 4: Group Audit

Group Testing Method Use Cases:

Some of the methods that can be used to test the use cases is to use these commands: `touch` to create a file, `nano` to write content in the file, `cat` to view the content of a file and `rm` to delete a file. When these commands were used by the authorized users, the users should be able to run the command without any problem as they have the permission for it. However, if an unauthorized user tries those commands to edit a file that they don't have the permission for, the terminal will output a "permission denied" message. (GeeksforGeeks, 2024)

Test case	Authorized User (Success)	Figure	Unauthorized User (Failure)	Figure
Create a new file	`touch <file>`	Figure 38.1 and 38.2	`touch <file>` (should fail if no write permission)	Figure 39.1
Modify a file (Owned by user)	`nano <file>`	Figure 38.3	`nano <file>` (should fail if no write permission)	Figure 39.2 and 39.3
Modify a file (group owned)	`nano <file>`	Figure 38.4, and 38.5	`nano <file>` (should fail if not in group)	Figure 39.4
Delete a file	`rm <file>`	Figure 38. 6 and 38.7	`rm <file>` (should fail if not owned or write permission)	Figure 39.5
Delete the group share	`rm -r /fileshare`	Figure 38.8 and 38.9	`rm -r /fileshare`	Figure 39.6

Figure 37: Group Testing Method Use Case

Group testing Method Auth User Use cases:

The testing method was used for an authorized user. The user account that was used for testing is Alice and this user has access to the /shared/security directory.

```
alice@25aSP102:~$ touch /shared/security/securityTest1.txt
alice@25aSP102:~$
```

Figure 38.1: use `touch` command to create group's owned file

```
cyberstudent@25aSP102:~$ sudo ls -l /shared/security/
[sudo] password for cyberstudent:
total 0
-rw-rw-r-- 1 alice Security 0 Feb 20 11:06 securityTest1.txt
-rw-r--r-- 1 root  root     0 Jan 24 13:03 security.txt
cyberstudent@25aSP102:~$
```

Figure 38.2 verified the file is created

```
alice@25aSP102:~$ nano alice.txt
alice@25aSP102:~$
```

Figure 38.3: Modified user's owned file

```
alice@25aSP102:/shared/security$ nano securityTest1.txt
alice@25aSP102:/shared/security$
```

Figure 38.4: Modified group's owned file

```
alice@25aSP102:/shared/security$ cat securityTest1.txt
Hello
alice@25aSP102:/shared/security$
```

Figure 38.5: Display the content of the file

```
alice@25aSP102:/shared/security$ rm securityTest1.txt
alice@25aSP102:/shared/security$
```

Figure 38.6: Deleted a file

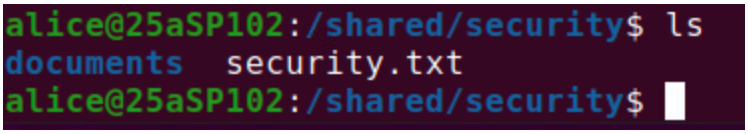

Figure 38.7: File inside the security shared folder

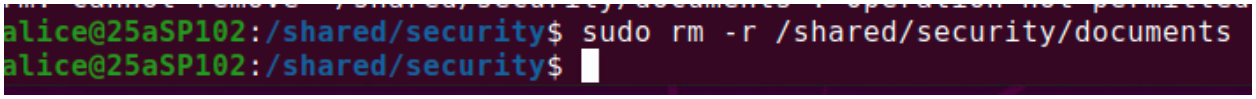

Figure 38.8: Deleted the group share folder

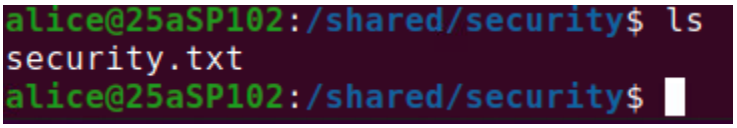

Figure 38.9: List of file in current security directory

Group testing Method Unauth User Misuse cases:

To test unauthenticated user misuse case, the same commands were used to test if an authorized user can edit the files and the terminal will show “permission denied” message when an unauthorized user tries to do so. Victor’s account was used to test if he can create a file inside the security shared directory.

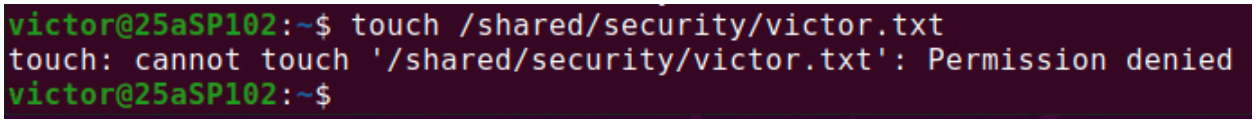

Figure 39.1: Permission denied message appear

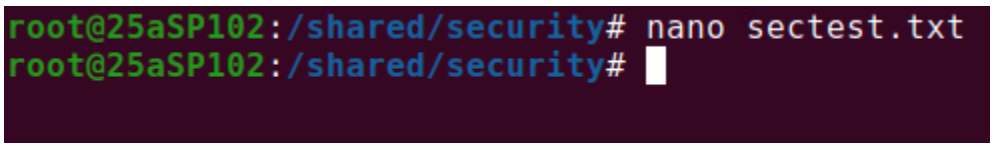

Figure 39.2: Attempt to modify Alice’s file

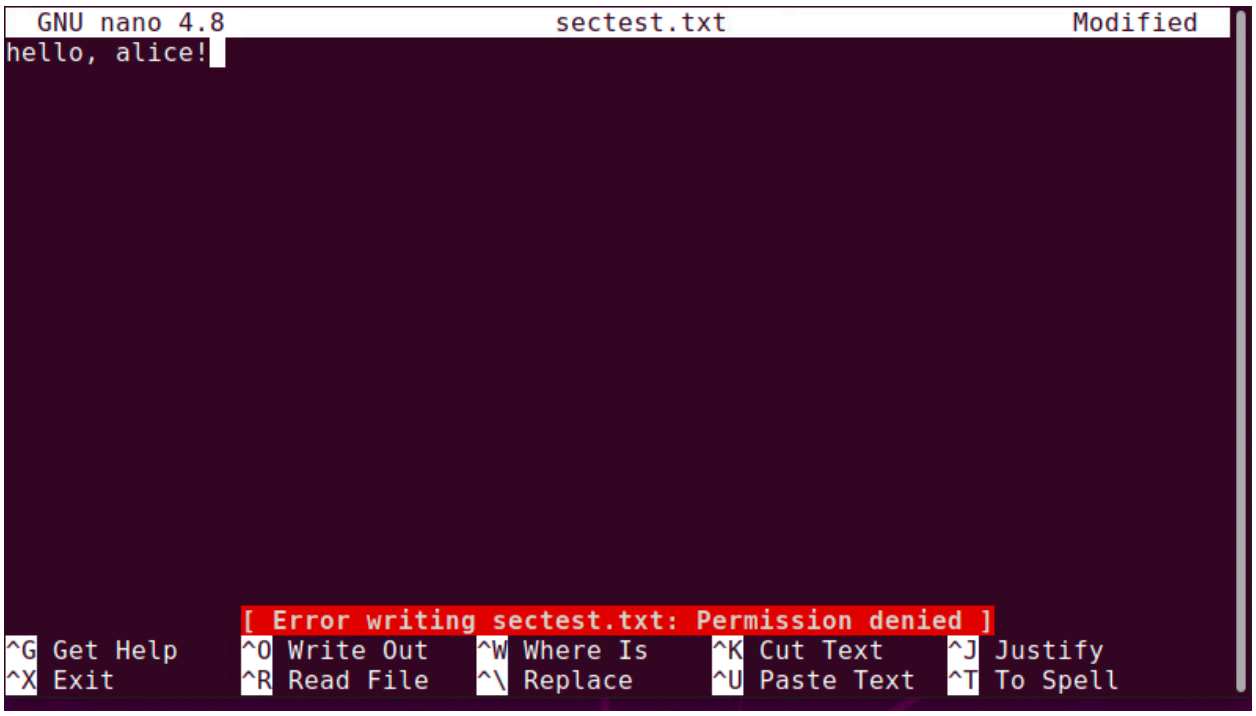

Figure 39.3 unable to write in a file where there is no write permission

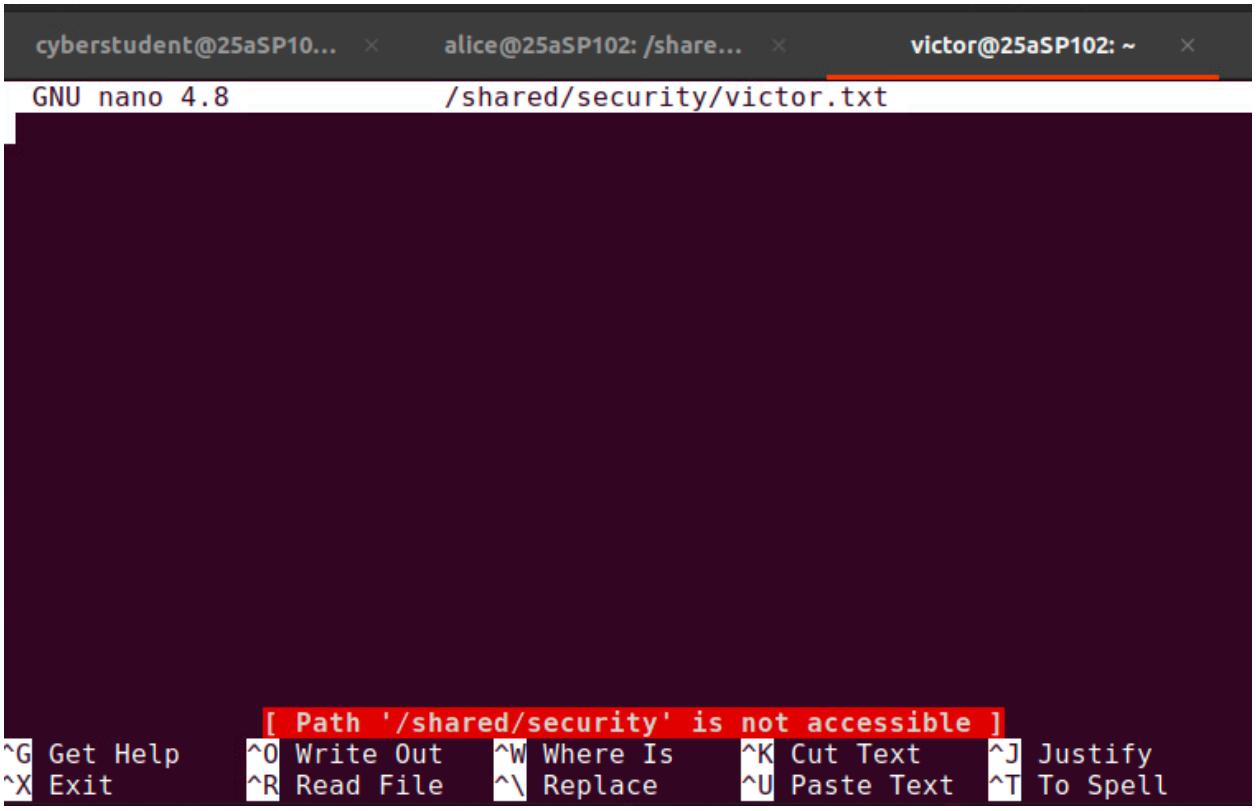

Figure 39.4: permission denied to modify the file

```
victor@25aSP102:~$ rm /shared/security/security.txt
rm: cannot remove '/shared/security/security.txt': Permission denied
victor@25aSP102:~$
```

Figure 39.5: permission denied to delete a file

```
victor@25aSP102:~$ rm -r /shared/security
rm: descend into write-protected directory '/shared/security'? y
rm: remove write-protected directory '/shared/security'? y
rm: cannot remove '/shared/security': Permission denied
victor@25aSP102:~$
```

Figure 39.6: permission denied to delete the group share

Question 5: Root Account

The first step to disable login for root users is writing the command `sudo passwd -l root`. This prevents direct logins by disabling the root password.

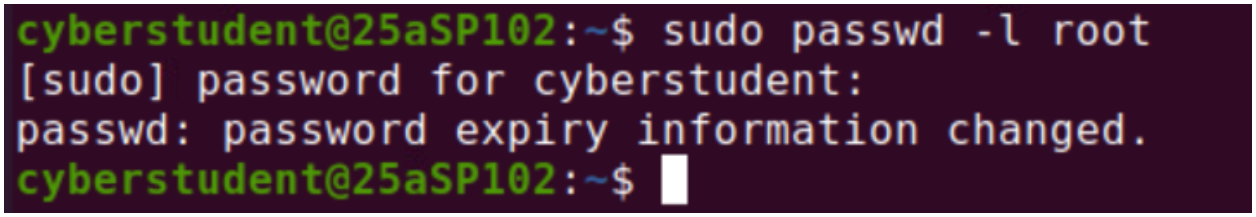

Figure 40: shows the command to lock the root account

The next step is to re-enable login by the root users by writing the command `sudo passwd root`. This will allow direct login, where then it prompts users to enter a new password.

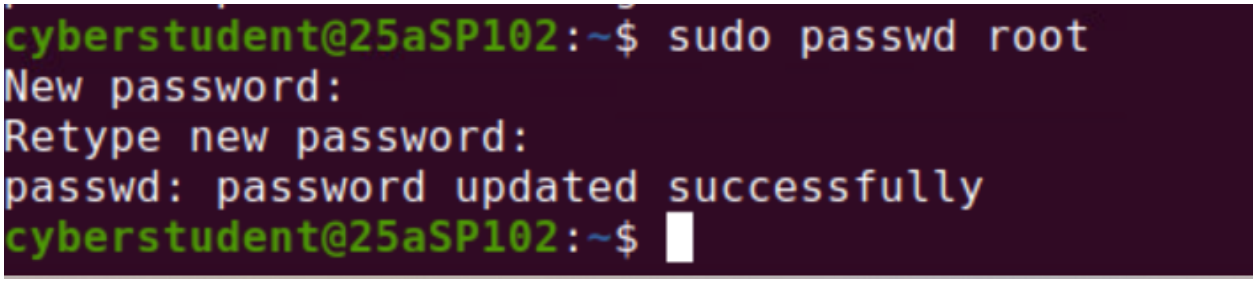

Figure 41: shows the command to unlock the root account and set a new password

As in figure 41, disabling the root account first is to prevent direct login as root by modifying its entry in `/etc/shadow`. When prompted to enter a new password in figure 41, Hands has specified the root account password as "`Sue, do you pseudo sew? So does doe.`" To know whether the account is disabled or enabled, is by looking at the output in the password field. If the root account is enabled, its password field will contain a hashed password instead of `!` or `*`.

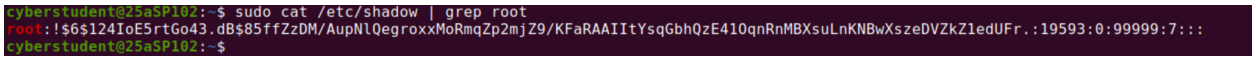

Figure 42: shows the root account is disabled as it contains `!`

Figure 43: shows the root account is now enabled as it doesn't have !

Figure 44: shows that root login is enabled by switching to the root user

Question 6: /etc/passwd & /etc/shadow

When the command `/etc/passwd` is entered in terminal, the output is shown in Figure 45 below:

```
cyberstudent@25aSP102:~$ sudo tail /etc/passwd
[sudo] password for cyberstudent:
fwupd-refresh:x:129:136:fwupd-refresh user,,,:/run/systemd:/usr/sbin/nologin
alice:x:1001:1001:Alice,1,1234567890,1234567890,1234567890:/home/alice:/bin/bash
bob:x:1002:1002:Bob,2,1234567891,1234567891,1234567891:/home/bob:/bin/bash
carol:x:1003:1003:Carol,3,1234567892,1234567892,1234567892:/home/carol:/bin/bash
eve:x:1004:1004:Eve,4,1234567893,1234567893,1234567893:/home/eve:/bin/bash
malory:x:1005:1005:Malory,5,1234567894,1234567894,1234567894:/home/malory:/bin/bash
trent:x:1006:1006:Trent,6,1234567895,1234567895,1234567895:/home/trent:/bin/bash
walter:x:1007:1007:Walter,7,1234567896,1234567896,1234567896:/home/walter:/bin/bash
victor:x:1008:1008:Victor,8,1234567897,1234567897,1234567897:/home/victor:/bin/bash
hands:x:1009:1009:Hands,9,1234567898,1234567898,1234567898:/home/hands:/bin/bash
cyberstudent@25aSP102:~$
```

Figure 45: the output for /etc/passwd

The format of the output:
username:password placeholder:UserID:GroupID: User ID Info: home directory: shell

username	Password placeholder	UserID	GroupID	User Info	Home directory	shell
alice	x	1001	1001	Alice,1,1234567890,1234567890,1234567890	/home/alice	/bin/sh

Figure 46: Understanding of `/etc/passwd`

Fields are separated by colons, ":". A while ago, when linux was not secure, the password would also be stored in this file, but since then has been migrated to /etc/shadow as a hash. (GeeksforGeeks, 2021) Due to this, it has a placeholder as "x" in where it used to be. Each user has a unique UserID, and they also have a main group they are in. User info is a field for arbitrary text, like notes for a person, such as their full name. Alice does not have anything in this field because we did not input this information when creating her account. Next is the home directory, which is simply where the user starts when they first login to the system (Anne, 2016). Lastly, is the shell to use within the terminal on their account, which in this case was automatically set to /bin/sh upon user creation.

When the command `/etc/shadow` is entered in terminal, the output is shown below:

Figure 47: output for /etc/shadow

username	Password hash	Last password change	Minimum password age	Maximum password age	Password warning period	Password inactivity period	Account expiration date
alice	6Pm7RtJyFrz6fhWC9$vpp499oVq6CnTgIs4HAwkzbrZlg/wUvx6wImsNRVG3AqKo6gAXQbN97slEvbBLrV0AWqVpQkm x.UihTFhD7VNO	20119	0	99999	7	-	-

Figure 48: Understanding of `/etc/shadow`

The first and second fields include the user's username and password hash, in that order. The third field indicates the last time the password was updated in unix time, or days since January 1, 1970. The user must wait at least four days before changing their password again. After setting a new password, the user has 5 days until it expires and must reset it upon login. The user is prompted to update their password six days before it expires. The account is automatically deactivated seven days after the password expires. The password will expire on the 8th, expressed in Unix time (days from January 1, 1970). (Gite, 2024)

Question 7: Create Character Devices

Explanation device permission:

`/dev/random` is used to generate random numbers, where it will be denied if the entropy pool is empty or does not indicate sufficient unpredictability. (Shamar, 2023). From the figure 31, `/dev/random` has 444 permissions which is a read only permission `(r- -r- - r - -)` for all users. This is essential to ensure that the random number that generated is not manipulated by any unintended processes.

On the other hand, `/dev/tty` is used to represent the terminal for the current process, which displays the terminal associated with the current SSH session (Ondara, 2023) has the 666 permissions which is read and write (`rw-rw-rw`) permission for all users. `/dev/tty` is owned by `root:tty` for limited access, hence, it must be writable to allow communication between users and terminal.

Question 8: Chroot Jail Testing

User jailed method and evidence:

To allow certain users to work from home via ssh connection, the authorized user should be part of a new group. Therefore, a group called remoteusers was made and command used was `sudo groupadd [group name]`

```
cyberstudent@25aSP102:/detention/usr/bin$ sudo groupadd remoteusers
cyberstudent@25aSP102:/detention/usr/bin$ sudo usermod -aG remoteusers alice
cyberstudent@25aSP102:/detention/usr/bin$ sudo usermod -aG remoteusers bob
cyberstudent@25aSP102:/detention/usr/bin$ nano /etc/ssh/sshd_config
cyberstudent@25aSP102:/detention/usr/bin$ sudo !!
sudo nano /etc/ssh/sshd_config
cyberstudent@25aSP102:/detention/usr/bin$ sudo systemctl restart sshd
cyberstudent@25aSP102:/detention/usr/bin$
```

Figure 49: shows commands to create a group for remote users and open the SSH file

```
cyberstudent@25aSP102:/$ sudo gpasswd -d bob remoteusers
Removing user bob from group remoteusers
cyberstudent@25aSP102:/$ sudo usermod -aG remoteusers carol
cyberstudent@25aSP102:/$
```

Figure 50: shows a command to add Carol as a remote user

Next, we created the root directory for the jail, which will be located at `/detention`.

```
cyberstudent@25aSP102:/$ sudo mkdir /detention
[sudo] password for cyberstudent:
Sorry, try again.
[sudo] password for cyberstudent:
cyberstudent@25aSP102:/$
```

Figure 51: creating detention directory in root directory

```
cyberstudent@25aSP102:/$ ls
bin   cdrom      dev  home  lib32  libx32      media  opt   root  sbin   snap  swapfile  tmp  var
boot  detention  etc  lib   lib64  lost+found  mnt    proc  run   shared srv   sys       usr
cyberstudent@25aSP102:/$
```

Figure 52: shows detention directory has been created

To complete the setup of a functional chroot jail for Carol and Alice, we begin by creating the necessary directory structure to replicate the root filesystem within the jail.

First, we create the base directory for the jail and replicate essential directories such as `/dev, /usr, /bin, /lib`, and `/etc` inside the jail. As the path `/detention/usr` need to have a subdirectory called bin, so the full path will be `/detention/usr/bin`

Figure 53: creating necessary subdirectories

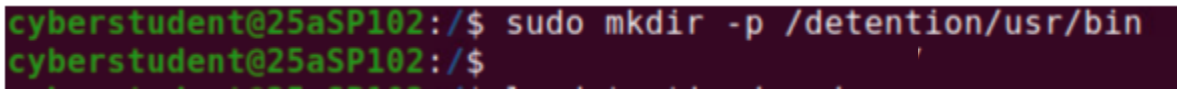

Figure 54: creating subdirectory bin

Next, the ownership of all these directories has been set to be owned by root to ensure security.

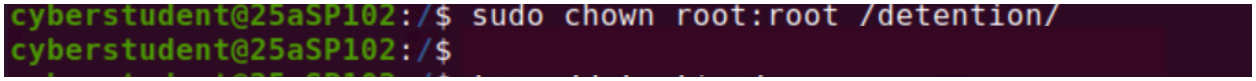

Figure 55: directories owned by root

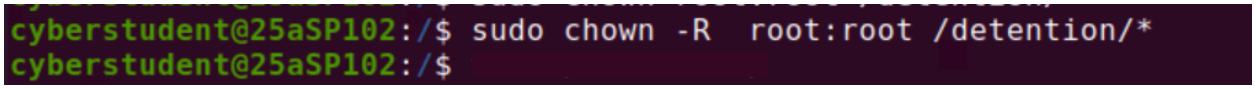

Figure 56: directories owned by root

To confirm that the directory structure is correct, we use the tree command : `tree /detention/ .` The tree command in Linux is used to display the directory structure of a path or the entire file system in a tree-like format. It lists directories, subdirectories, and files in a hierarchical manner. (Geekforgeeks, 2024) This sets up the foundation for a chroot jail in `/detention`, isolating SSH users within this restricted environment.

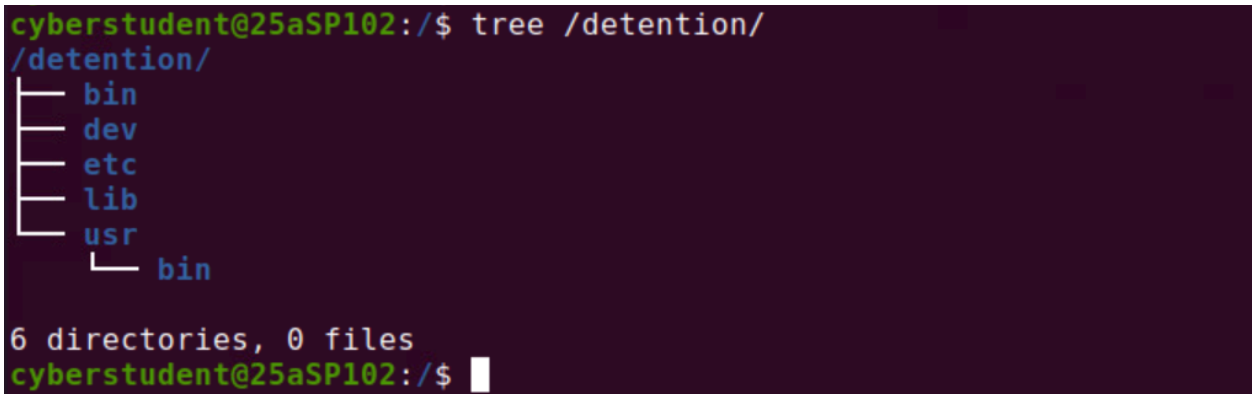

Figure 57: tree command showing correct root file structure in detention directory

Character devices in Linux allow processes to interface directly with hardware without the need for buffering. These are required for the chroot prison to work normally. (Geeksforgeeks,2024). Hence, Alice and Bob must use the `mknod` command to create these devices in `/detention/dev/`.

```
cyberstudent@25aSP102:/$ sudo mknod -m 666 /detention/dev/null c 1 3
cyberstudent@25aSP102:/$
```

Figure 58.1 : create character devices inside the chroot jail

```
cyberstudent@25aSP102:/$ sudo mknod -m 622 /detention/dev/zero c 1 5
cyberstudent@25aSP102:/$
```

Figure 58.2 : create character devices inside the chroot jail

```
cyberstudent@25aSP102:/$ sudo mknod -m 666 /detention/dev/tty c 5 0
cyberstudent@25aSP102:/$
```

Figure 58.3 : create character devices inside the chroot jail

```
cyberstudent@25aSP102:/$ sudo mknod -m 444 /detention/dev/random c 1 8
cyberstudent@25aSP102:/$
```

Figure 58.4 : create character devices inside the chroot jail

```
cyberstudent@25aSP102:/$ sudo mknod -m 444 /detention/dev/urandom c 1 9
cyberstudent@25aSP102:/$
```

Figure 58.5 : create character devices inside the chroot jail

```
cyberstudent@25aSP102:/$ sudo chown root:tty /detention/dev/tty
cyberstudent@25aSP102:/$
```

Figure 58.6 : create character devices inside the chroot jail

To ensure that users in the chroot jail have access to necessary system utilities while maintaining security, Alice and Bob must copy essential files from the regular filesystem into the jail. The `/etc` directory inside the chroot jail needs key configuration files for system operations (Geeksforgeeks,2024). Following commands can be seen in figures below.

```
cyberstudent@25aSP102:/$ cd detention/etc
cyberstudent@25aSP102:
```

Figure 59.1: populate the `/detention/etc` directory with key files

```
cyberstudent@25aSP102:/detention/etc$ sudo cp /etc/ld.so.cache .
cyberstudent@25aSP102:/detention/etc$
```

Figure 59.2: populate the `/detention/etc` directory with key files

```
cyberstudent@25aSP102:/detention/etc$ sudo cp /etc/ld.so.conf .
cyberstudent@25aSP102:/detention/etc$
```

Figure 59.3: populate the `/detention/etc` directory with key files

```
cyberstudent@25aSP102:/detention/etc$ sudo cp /etc/nsswitch.conf .
cyberstudent@25aSP102:/detention/etc$
```

Figure 59.4: populate the `/detention/etc` directory with key files

```
cyberstudent@25aSP102:/detention/etc$ sudo cp /etc/hosts .
cyberstudent@25aSP102:/detention/etc$
```

Figure 59.5: populate the `/detention/etc` directory with key files

Next, copying essential binaries into the `/detention` directory. In order to allow users to do basic functions, the jail must have the `ls` command and a shell (bash).

```
cyberstudent@25aSP102:/detention/etc$ cd /detention/bin/
cyberstudent@25aSP102:/detention/bin$
```

Figure 60.1: Copying basic system utilities to `/detention/bin`

```
cyberstudent@25aSP102:/detention/bin$ sudo cp /bin/ls .
cyberstudent@25aSP102:/detention/bin$
```

Figure 60.2: Copying basic system utilities to `/detention/bin`

```
cyberstudent@25aSP102:/detention/bin$ sudo cp /bin/bash .
cyberstudent@25aSP102:/detention/bin$
```

Figure 60.3: Copying basic system utilities to `/detention/bin`

Many system utilities rely on shared libraries. Instead of manually copying them, a script will automate this process.This command lists all shared libraries required by `/bin/ls`. These libraries are copied into the jail in the figure below. `l2chroot` is a script that helps copy necessary shared libraries for chroot utilities and `chmod 755` ensures it has execute permissions (Geeksforgeeks,2024).

```
cyberstudent@25aSP102:/detention/bin$ ldd /bin/ls
        linux-vdso.so.1 (0x00007ffec4bde000)
        libselinux.so.1 => /lib/x86_64-linux-gnu/libselinux.so.1 (0x00007f4f67cb9000)
        libc.so.6 => /lib/x86_64-linux-gnu/libc.so.6 (0x00007f4f67ac7000)
        libpcre2-8.so.0 => /lib/x86_64-linux-gnu/libpcre2-8.so.0 (0x00007f4f67a36000)
        libdl.so.2 => /lib/x86_64-linux-gnu/libdl.so.2 (0x00007f4f67a30000)
        /lib64/ld-linux-x86-64.so.2 (0x00007f4f67d1e000)
        libpthread.so.0 => /lib/x86_64-linux-gnu/libpthread.so.0 (0x00007f4f67a0d000)
cyberstudent@25aSP102:/detention/bin$
```

Figure 61: shows the command to check dependencies for `ls`

```
cyberstudent@25aSP102:/detention/usr/bin$ cd /detention/usr/bin/
cyberstudent@25aSP102:/detention/usr/bin$
```

Figure 62.1: shows the commands to copy the libraries into the jail

```
cyberstudent@25aSP102:/detention/usr/bin$ sudo cp /home/cyberstudent/l2chroot .
cyberstudent@25aSP102:/detention/usr/bin$
```

Figure 62.2: shows the commands to copy the libraries into the jail

```
cyberstudent@25aSP102:/detention/usr/bin$ sudo chmod 755 l2chroot
cyberstudent@25aSP102:/detention/usr/bin$
```

Figure 62.3: shows the commands to copy the libraries into the jail

As we need to change `/jailed` to `/detention`, the command `sudo nano l2chrooot` is used and the line `BASE = "/jailed"` is changed to `BASE = "/detention"`. The file is then saved. The script is then executed to copy dependencies for the essential utilities. This ensures that all required shared libraries are copied into the jail, allowing the `ls` and `bash` commands to function properly inside `/detention`. (Geeksforgeeks,2024).

```
cyberstudent@25aSP102:/detention/usr/bin$ sudo nano l2chroot
cyberstudent@25aSP102:/detention/usr/bin$
```

Figure 63: modifying file to reference to chroot directory

```
cyberstudent@25aSP102:/detention/usr/bin$ sudo /detention/usr/bin/l2chroot /bin/ls
Copying shared files/libs to /detention...
Copying /lib64/ld-linux-x86-64.so.2 /detention/lib64...
cyberstudent@25aSP102:/detention/usr/bin$
```

Figure 64.1: Chroot execution

```
cyberstudent@25aSP102:/detention/usr/bin$ sudo /detention/usr/bin/l2chroot /bin/bash
Copying shared files/libs to /detention...
cyberstudent@25aSP102:/detention/usr/bin$
```

Figure 64.2: Chroot execution

Next, to enforce the chroot jail for remote SSh users, Alice and Bob modify the SSH daemon configuration. Once the SSh configuration file was opened, the `LogLevel INFO` was changed to `LogLevel VERBOSE` and these lines were added at the bottom of the file: `Match group remoteusers, ChrootDirectory /detention/ .` The `Match group remoteusers` directive ensures only users in the remoteusers group (Carol & Alice) are affected and the `ChrootDirectory /detention/` forces these users into the jail upon SSH login. (Geeksforgeeks,2023).

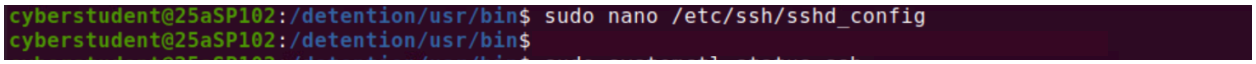

Figure 65: shows the command to open the SSH file

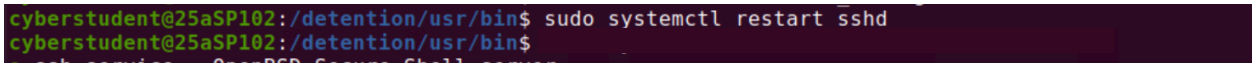

Figure 66: shows the command to restart the SSH file

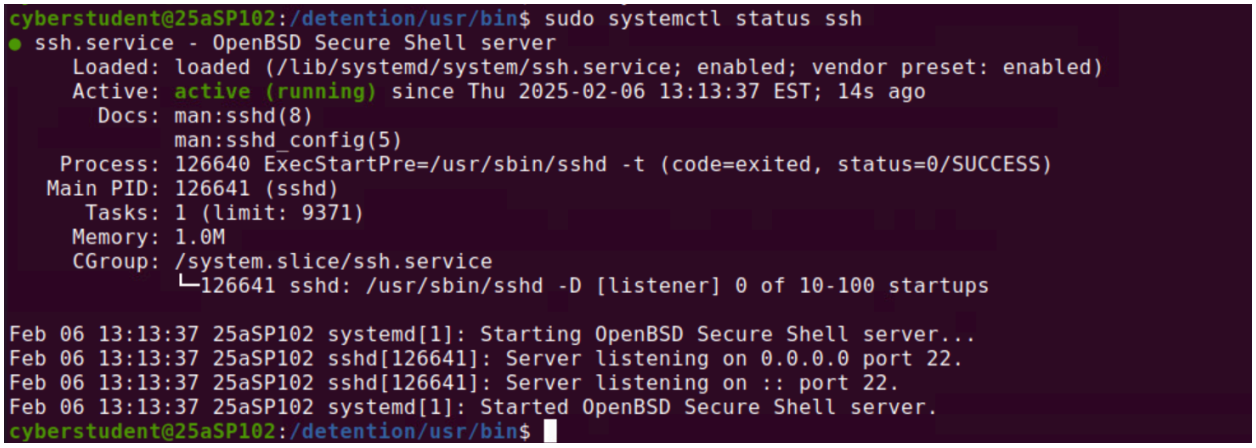

Figure 67: shows the command to check the SSH status

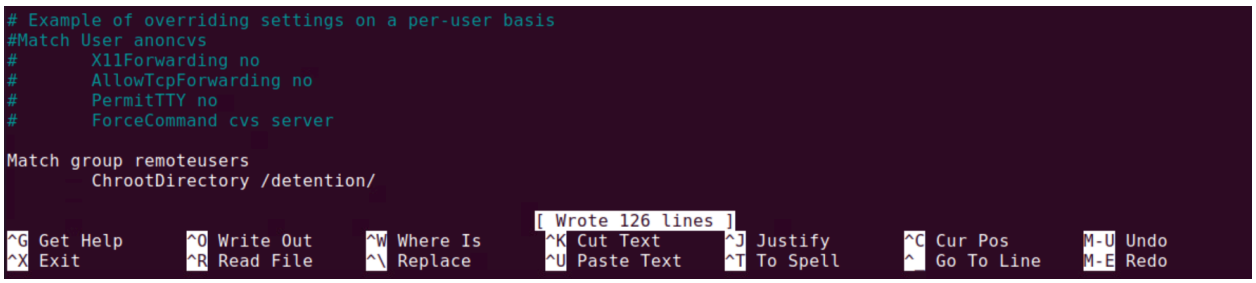

Figure 68: shows the following lines inside the SSH file

To confirm that Carol and Alice are jailed when logging in over SSH, the following tests has been performed:

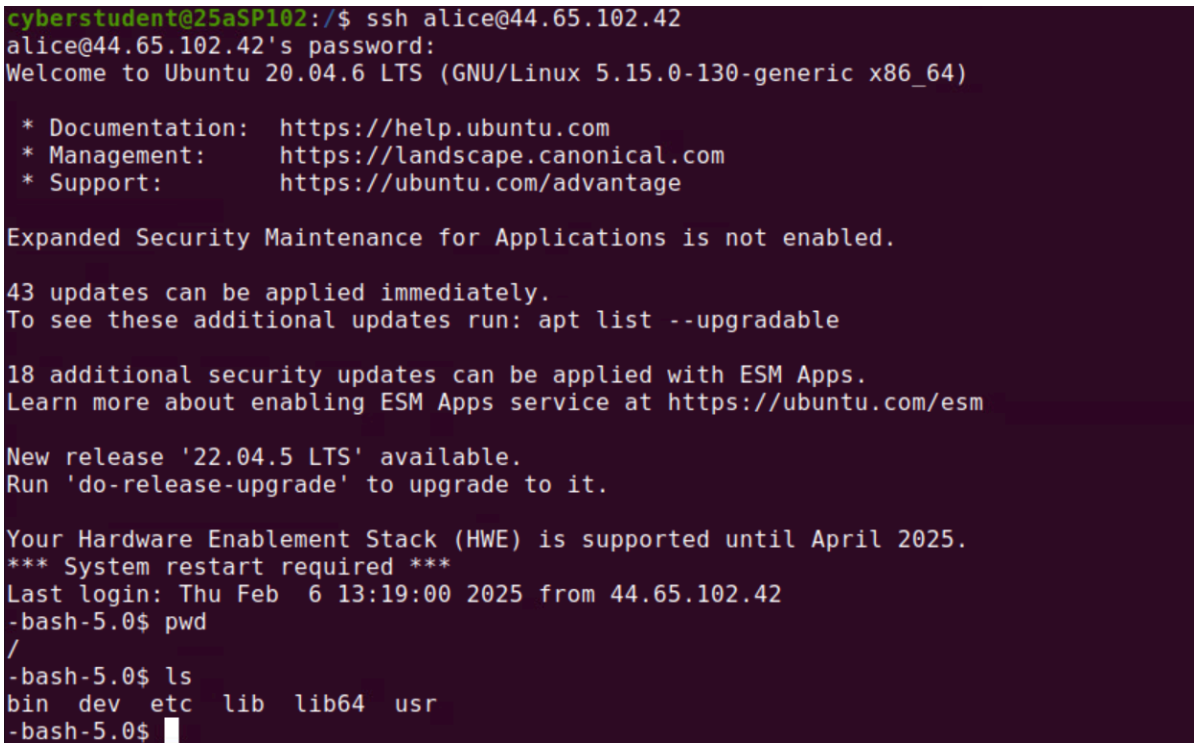

Figure 69: shows that Alice has successfully jailed when logging in over SSH

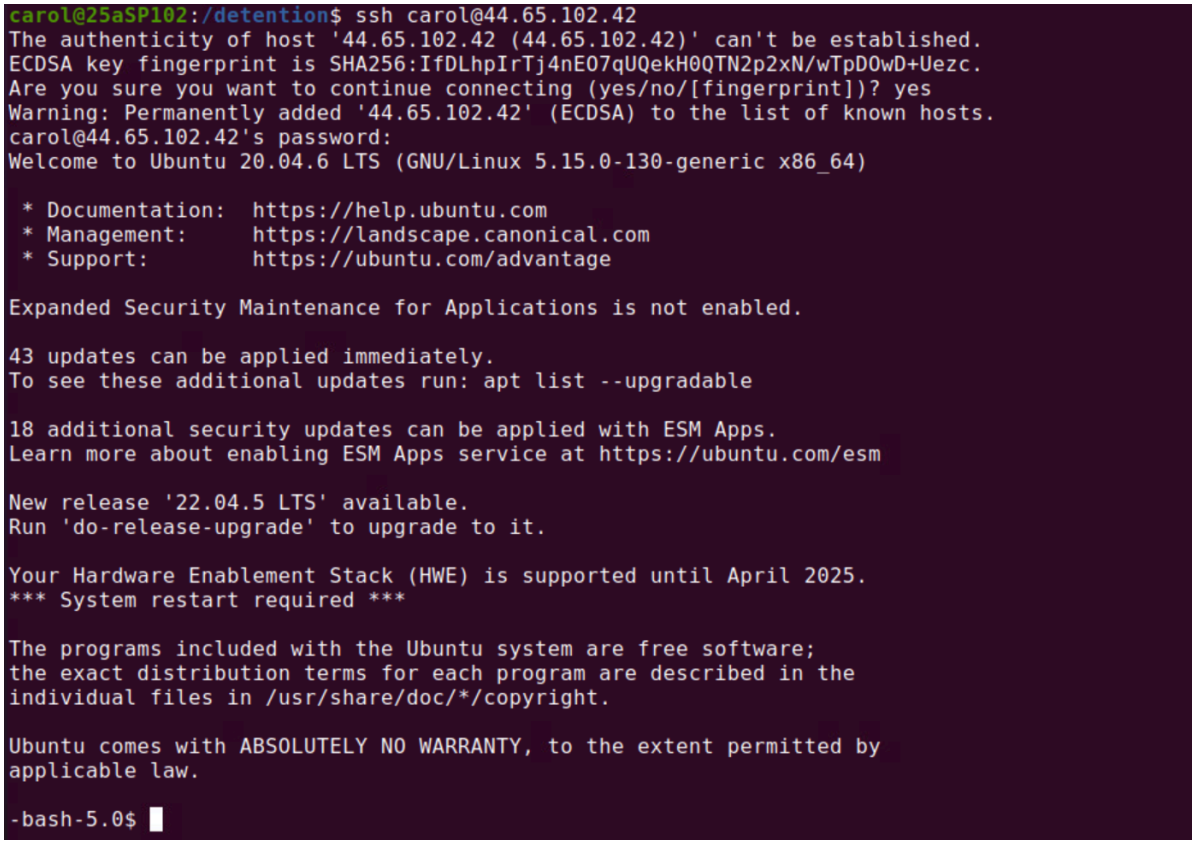

Figure 70: shows that Carol has successfully jailed when logging in over SSH

Question 9: Chroot Jail: Making the Jail Nice

```
cyberstudent@25aSP102:/detention$ sudo !!
sudo mkdir -p /detention/home/alice
cyberstudent@25aSP102:/detention$
```

Figure 71: add home directories for Alice inside the jail

```
cyberstudent@25aSP102:/detention$ sudo mkdir -p /detention/home/carol
cyberstudent@25aSP102:/detention$
```

Figure 72: add home directories for Carol inside the jail

```
cyberstudent@25aSP102:/detention$ sudo mkdir -p /detention/shared
cyberstudent@25aSP102:/detention$
```

Figure 73: add shared workspace inside the jail

```
cyberstudent@25aSP102:/detention$ sudo chown root:root /detention/home
cyberstudent@25aSP102:/detention$
```

Figure 74.1: ensure root owns the jail and gives permission to user

```
cyberstudent@25aSP102:/detention$ sudo chown alice:alice /detention/home/alice/
cyberstudent@25aSP102:/detention$
```

Figure 74.2: ensure root owns the jail and gives permission to user

```
cyberstudent@25aSP102:/detention$ sudo chown carol:carol /detention/home/carol/
cyberstudent@25aSP102:/detention$
```

Figure 74.3: ensure root owns the jail and gives permission to user

```
cyberstudent@25aSP102:/detention$ sudo chown root:remoteusers /detention/shared/
cyberstudent@25aSP102:/detention$
```

Figure 74.4: ensure root owns the jail and gives permission to user

```
cyberstudent@25aSP102:/detention$ sudo !!
sudo chmod 770 /detention/shared/
cyberstudent@25aSP102:/detention$
```

Figure 74.5: ensure root owns the jail and gives permission to user

```
cyberstudent@25aSP102:/detention$ sudo cp /bin/vim /detention/bin/
cyberstudent@25aSP102:/detention$
```

Figure 75.1: shows the commands to copy `vim`

```
cyberstudent@25aSP102:/detention$ sudo cp /usr/bin/tail /detention/usr/bin/
cyberstudent@25aSP102:/detention$
```

Figure 75.2: shows the commands to copy `tail`

```
cyberstudent@25aSP102:/detention$ sudo cp /bin/echo /detention/bin/
cyberstudent@25aSP102:/detention$
```

Figure 75.3: shows the commands to copy `echo`

```
cyberstudent@25aSP102:/detention$ ldd /bin/vim
        linux-vdso.so.1 (0x00007fff5dd34000)
        libm.so.6 => /lib/x86_64-linux-gnu/libm.so.6 (0x00007fd694f82000)
        libtinfo.so.6 => /lib/x86_64-linux-gnu/libtinfo.so.6 (0x00007fd694f52000)
        libselinux.so.1 => /lib/x86_64-linux-gnu/libselinux.so.1 (0x00007fd694f27000)
        libcanberra.so.0 => /lib/x86_64-linux-gnu/libcanberra.so.0 (0x00007fd694f14000)
        libacl.so.1 => /lib/x86_64-linux-gnu/libacl.so.1 (0x00007fd694f09000)
        libgpm.so.2 => /lib/x86_64-linux-gnu/libgpm.so.2 (0x00007fd694d03000)
        libdl.so.2 => /lib/x86_64-linux-gnu/libdl.so.2 (0x00007fd694cfb000)
        libpython3.8.so.1.0 => /lib/x86_64-linux-gnu/libpython3.8.so.1.0 (0x00007fd6947a5000)
        libpthread.so.0 => /lib/x86_64-linux-gnu/libpthread.so.0 (0x00007fd694782000)
        libc.so.6 => /lib/x86_64-linux-gnu/libc.so.6 (0x00007fd694590000)
        /lib64/ld-linux-x86-64.so.2 (0x00007fd6953bc000)
        libpcre2-8.so.0 => /lib/x86_64-linux-gnu/libpcre2-8.so.0 (0x00007fd6944ff000)
        libvorbisfile.so.3 => /lib/x86_64-linux-gnu/libvorbisfile.so.3 (0x00007fd6944f4000)
        libtdb.so.1 => /lib/x86_64-linux-gnu/libtdb.so.1 (0x00007fd6944d8000)
        libltdl.so.7 => /lib/x86_64-linux-gnu/libltdl.so.7 (0x00007fd6944cd000)
        libexpat.so.1 => /lib/x86_64-linux-gnu/libexpat.so.1 (0x00007fd69449f000)
        libz.so.1 => /lib/x86_64-linux-gnu/libz.so.1 (0x00007fd694483000)
        libutil.so.1 => /lib/x86_64-linux-gnu/libutil.so.1 (0x00007fd69447e000)
        libvorbis.so.0 => /lib/x86_64-linux-gnu/libvorbis.so.0 (0x00007fd69444e000)
        libogg.so.0 => /lib/x86_64-linux-gnu/libogg.so.0 (0x00007fd694441000)
cyberstudent@25aSP102:/detention$
```

Figure 76.1: shows the command to identify the necessary shared libraries

```
cyberstudent@25aSP102:/detention$ ldd /usr/bin/tail
        linux-vdso.so.1 (0x00007ffd63fa6000)
        libc.so.6 => /lib/x86_64-linux-gnu/libc.so.6 (0x00007fa18917e000)
        /lib64/ld-linux-x86-64.so.2 (0x00007fa189398000)
cyberstudent@25aSP102:/detention$
```

Figure 76.2: shows the commands to identify the necessary shared libraries

```
cyberstudent@25aSP102:/detention$ ldd /bin/echo
        linux-vdso.so.1 (0x00007fff1a3b1000)
        libc.so.6 => /lib/x86_64-linux-gnu/libc.so.6 (0x00007f31bb430000)
        /lib64/ld-linux-x86-64.so.2 (0x00007f31bb642000)
cyberstudent@25aSP102:/detention$
```

Figure 76.3: shows the commands to identify the necessary shared libraries

The method for adding the new binaries and their dependencies to the jail is by determining the required shared libraries using the `ldd` command. The `ldd` command in Figure 76.1 prints the shared libraries by each binary. (Stocker, 2025)

```
cyberstudent@25aSP102:/detention$ sudo /detention/usr/bin/l2chroot /bin/vim
Copying shared files/libs to /detention...
cyberstudent@25aSP102:/detention$
```

Figure 77.1: shows the command to copy those libraries into the chroot jail

```
cyberstudent@25aSP102:/detention$ sudo /detention/usr/bin/l2chroot /usr/bin/tail
Copying shared files/libs to /detention...
cyberstudent@25aSP102:/detention$
```

Figure 77.2: shows the command to copy those libraries into the chroot jail

```
cyberstudent@25aSP102:/detention$ sudo /detention/usr/bin/l2chroot /bin/echo
Copying shared files/libs to /detention...
cyberstudent@25aSP102:/detention$
```

Figure 77.3: shows the command to copy those libraries into the chroot jail

```
cyberstudent@25aSP102:/detention$ sudo mount --rbind /home/alice/ /detention/home/alice/
cyberstudent@25aSP102:/detention$
```

Figure 78: shows the commands to mount alice's home directory using `-rbind`.

```
cyberstudent@25aSP102:/detention$ findmnt /detention/home/alice
TARGET                SOURCE                 FSTYPE OPTIONS
/detention/home/alice /dev/sda2[/home/alice] ext4   rw,relatime,errors=remount-ro
cyberstudent@25aSP102:/detention$
```

Figure 79: shows the commands that finds the mounts of the alice home directory is inside the jail.

The two commands in figure 78 and 79 are repeated for Carol.

```
cyberstudent@25aSP102:/detention$ sudo mount --rbind /shared/ /detention/shared/
cyberstudent@25aSP102:/detention$
```

Figure 80: mount the users group shares inside the jail

```
cyberstudent@25aSP102:/detention$ findmnt /detention/shared
TARGET             SOURCE              FSTYPE OPTIONS
/detention/shared /dev/sda2[/shared] ext4   rw,relatime,errors=remount-ro
cyberstudent@25aSP102:/detention$
```

Figure 81: finds the mount of the shared directory.

The `mount` command mounts the directory specified to a different location. We used the `--rbind` flag to “recursively bind” the sub-mounts of a directory. This was done as a preventative measure in case something went wrong during the process. (Marijan, 2023). This step ensures that both users can access their home directories and shared resources (Purdue IT, 2023), even though the jail restricts their access to other parts of the system. We also make sure that the mounted directories are properly set up so that Carol and Alice cannot access or modify each other’s files. We achieve this by setting the correct ownership and permissions on their respective home directories, ensuring each user can only read and write to their own directory.

Question 10: Chroot Jail: Nice Jail Testing

After setting up the jail, we perform thorough testing to confirm the setup is correct. We test SSH access to ensure both Carol and Alice can log in to the jail, and we verify that they are able to edit files within their own directories but cannot access or modify files in each other's directories. We also test the mounted shares and check that they are properly accessible. Finally, we ensure that all file operations, such as echoing text into files and viewing log files, are functioning as expected. (Purdue IT, 2023)

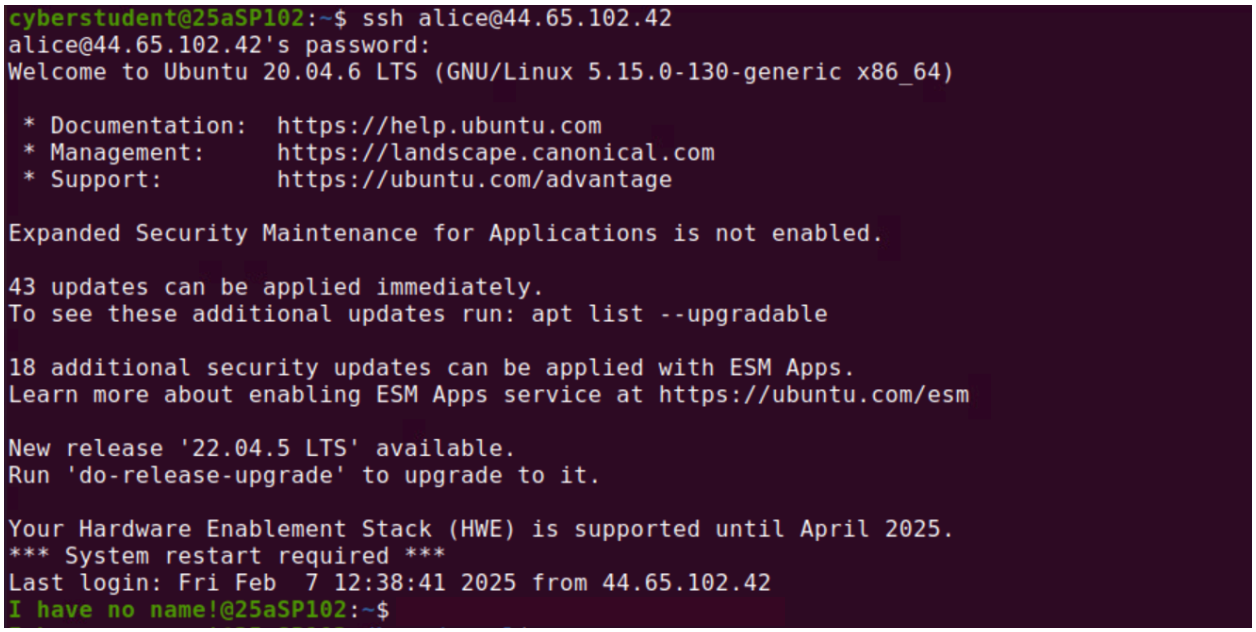

Figure 82: SSH into Alice

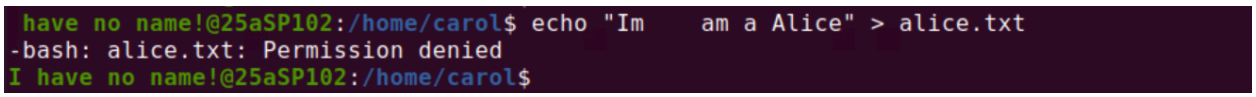

Figure 83: Alice doesn't have permission to write in Carol's directory.

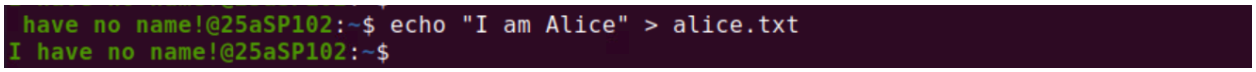

Figure 84: write inside Alice's text file

```
 have no name!@25aSP102:~$ tail alice.txc t
I am Alice
I have no name!@25aSP102:~$
```

Figure 85: Alice has the access to `tail` in hers directory

```
I have no name!@25aSP102:~$ vim alice.txt
"alice.txt" 1 line, 11 charactersI am Alice
~
~
~
~
~
~
~
```

Figure 86: shows that Alice has access to vim text editor

```
cyberstudent@25aSP102:~$ ssh carol@44.65.102.42
carol@44.65.102.42's password:
Welcome to Ubuntu 20.04.6 LTS (GNU/Linux 5.15.0-130-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

Expanded Security Maintenance for Applications is not enabled.

43 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

18 additional security updates can be applied with ESM Apps.
Learn more about enabling ESM Apps service at https://ubuntu.com/esm

New release '22.04.5 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Your Hardware Enablement Stack (HWE) is supported until April 2025.
*** System restart required ***
Last login: Fri Feb  7 12:32:38 2025 from 44.65.102.42
I have no name!@25aSP102:~$
```

Figure 87: SSH into Carol

```
 have no name!@25aSP102:/home/alice$ echo "I am Carol" > carol.txt
-bash: carol.txt: Permission denied
I have no name!@25aSP102:/home/alice$
```

Figure 88: Carol doesn't have permission to write in Alice’s directory.

```
 have no name!@25aSP102:/home/alice$ tail alice.txt
I am Alice
I have no name!@25aSP102:/home/alice$
```

Figure 89: Carol has permission to read

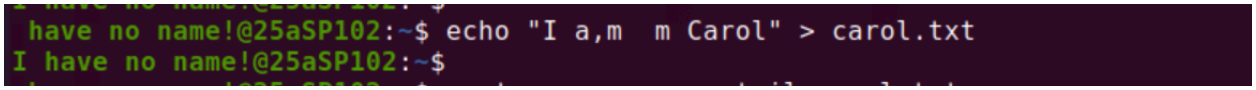

Figure 90: Carol has the access to `echo` in his directory

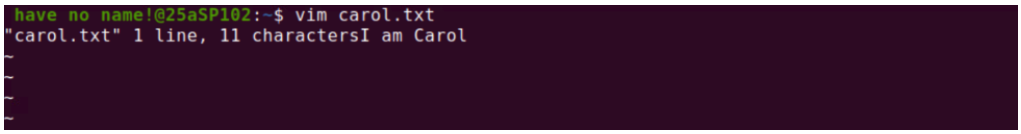

Figure 91: shows that carol has access to vim text editor

The chroot jail is now fully functional and secure, allowing Carol and Alice to work remotely via SSH with limited but necessary commands.

References

Abhishek. (2022, July 15). *How to Create a Folder in Ubuntu Command Line*. Learn Ubuntu. https://learnubuntu.com/create-folder/#:~:text=Let

Aleksic, M. (2019, April 16). *How to List Users in Linux with the Command Line {3 Easiest Ways}*. Knowledge Base by PhoenixNAP. https://phoenixnap.com/kb/how-to-list-users-linux

in. (2012, November 16). *Login as non-root user in terminal*. Ask Ubuntu. https://askubuntu.com/questions/217912/login-as-non-root-user-in-terminal#:~:text=Append%20-s%20/bin/bash%20to%20your%20su%20command%20or%20use%20chsh

GeeksforGeeks. (2023, October 10). How to Add User to a Group in Linux. GeeksforGeeks. Retrieved January 30, 2025, from https://www.geeksforgeeks.org/how-to-add-a-user-to-a-group-in-linux/

GeeksforGeeks. (2024, September 3). getent command in Linux with examples. GeeksforGeeks. Retrieved January 30, 2025, from https://www.geeksforgeeks.org/getent-command-in-linux-with-examples/

GeeksforGeeks. (2024, September 18). How to unlock a locked user account in linux? GeeksforGeeks. Retrieved January 31, 2025, from https://www.geeksforgeeks.org/how-to-unlock-a-locked-user-account-in-linux/

GeeksforGeeks. (2024, December 18). How to Set File Permissions in Linux. GeeksforGeeks. Retrieved January 30, 2025, from https://www.geeksforgeeks.org/set-file-permissions-linux/

Gite, V. (2024, September 16). *Understanding /etc/passwd File Format*. nixCraft. https://www.cyberciti.biz/faq/understanding-etcpasswd-file-format/

Gite, V. (2024, September 16). *Understanding /etc/shadow file format on Linux*. nixCraft. https://www.cyberciti.biz/faq/understanding-etcshadow-file/

How to Add and Remove Users on Ubuntu 20.04. (2020, August 22). Linuxize.com. https://linuxize.com/post/how-to-add-and-delete-users-on-ubuntu-20-04/

Manav014. (2023, December 22). How to add user in Linux | useradd command. GeeksforGeeks. https://www.geeksforgeeks.org/useradd-command-in-linux-with-examples/

NERSC. (n.d.). UNIX file permissions. NERSC Documentation. https://docs.nersc.gov/filesystems/unix-file-permissions/

Sethusubramanian. (2022, September 6). Chage command in Linux with examples. GeeksforGeeks. https://www.geeksforgeeks.org/chage-command-in-linux-with-examples/

Sharma, S. (2023, February 20). What are /Dev/random and /Dev/urandom in Linux? Linux Handbook. https://linuxhandbook.com/dev-random-urandom/

Structures of Directory in Operating System - GeeksforGeeks. (2018, October 31). GeeksforGeeks. https://www.geeksforgeeks.org/structures-of-directory-in-operating-system/

Ondara, W. (2023, May 24). What is /Dev/tty, /Dev/tty0, and /Dev/console in Linux. Linux: Difference Between /dev/tty, /dev/tty0, and /dev/console. https://www.tecmint.com/linux-tty-tty0-and-console/

Marijan, B. (2023, October 23). *Linux Mount Command with examples {+how to Unmount a file system}*. Knowledge Base by phoenixNAP. https://phoenixnap.com/kb/linux-mount-command

Stocker, S. H. (2025, February 18). *Using the LDD command on linux*. Network World. https://www.networkworld.com/article/970808/using-the-ldd-command-on-linux.html

Tree command in Linux with examples. (2024, December 30). GeeksforGeeks. https://www.geeksforgeeks.org/tree-command-unixlinux/

Purdue IT. (2023, November 15). Mounting Home Directory on a Linux Operating System. Purdue. https://engineering.purdue.edu/ECN/Support/KB/Docs/MountingHomeDirectory