

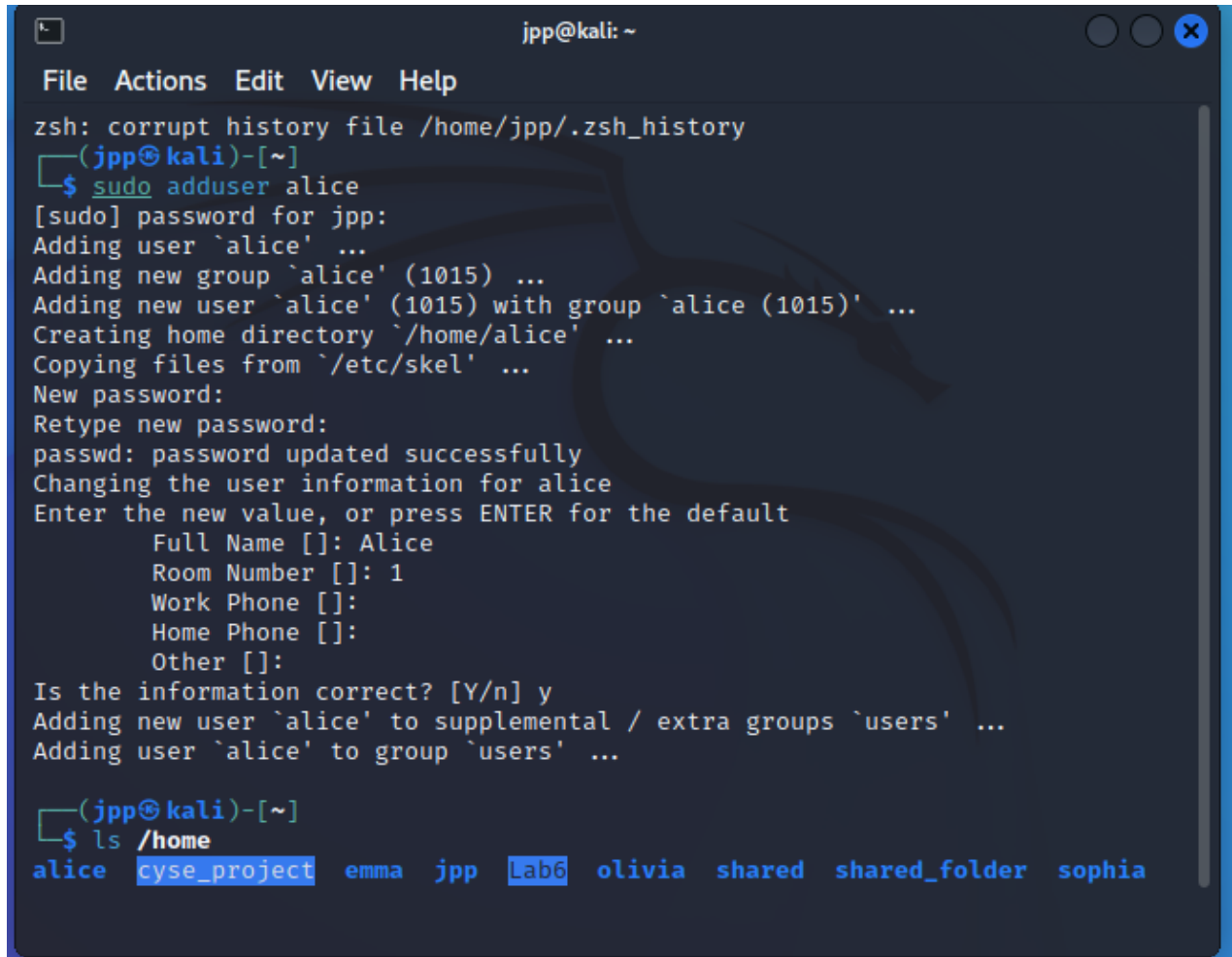
CYSE 270: Linux System for Cybersecurity

Assignment-9

Task A - Backup your system (Using crontab) [100 points]

Scenario: Performing system backup can be time-consuming, and the process is often overlooked. For this scenario,

1. (10 Points) Create a new user **Alice** (with home directory) and

A terminal window titled 'jpp@kali: ~' with a menu bar (File, Actions, Edit, View, Help). The terminal shows the execution of 'sudo adduser alice'. It prompts for a password, then displays the progress of adding the user, including creating a home directory and copying files. It then prompts for a new password and its retype. After confirming the user information (Full Name: Alice, Room Number: 1, etc.), it asks if the information is correct (y). Finally, it adds the user to the 'users' group. The prompt returns to '(jpp@kali)-[~]'. The next command is 'ls /home', which lists the following directories: 'alice', 'cyse_project', 'emma', 'jpp', 'Lab6', 'olivia', 'shared', 'shared_folder', and 'sophia'.

```
jpp@kali: ~
File Actions Edit View Help
zsh: corrupt history file /home/jpp/.zsh_history
(jpp@kali)-[~]
$ sudo adduser alice
[sudo] password for jpp:
Adding user `alice' ...
Adding new group `alice' (1015) ...
Adding new user `alice' (1015) with group `alice (1015)' ...
Creating home directory `/home/alice' ...
Copying files from `/etc/skel' ...
New password:
Retype new password:
passwd: password updated successfully
Changing the user information for alice
Enter the new value, or press ENTER for the default
    Full Name []: Alice
    Room Number []: 1
    Work Phone []:
    Home Phone []:
    Other []:
Is the information correct? [Y/n] y
Adding new user `alice' to supplemental / extra groups `users' ...
Adding user `alice' to group `users' ...

(jpp@kali)-[~]
$ ls /home
alice  cyse_project  emma  jpp  Lab6  olivia  shared  shared_folder  sophia
```

- 2.
3. (50 Points) Write a shell script that backups Alice's home directory by creating a tar file (tape archive), using the following steps:
 - i. Take 2 inputs with their values- your **MIDAS** name and **current date** (for example, midas=svatsa).

- ii. Create a variable named as **filename** that should be assigned the value as **MIDAS-date** (example output after executing the script would be like, **svatsa-2021.3.17-01.16.430**).
 - iii. Using **tar** command, create a tape archive for Alice's home directory (/home/Alice) and the **filename** created above (in step-2-ii). (Please learn about tar command in Linux for its usage)
- b. Move the tape archive file/tar file (created in step 2-iii) to /var/backups/ directory using correct command in linux.
- c. To optimize the disk usage, pick a compression algorithm (bz2, gzip, or xv) to compress the tar file you created in /var/backups/ in the previous step-2b.

```
File Actions Edit View Help
#!/bin/bash

read -p "Enter your MIDAS name: " midas
read -p "Enter the current date (YYYY.MM.DD): " date

filename="${midas}-${date}-${date +%H.%M.%S}"

tar -cvf "${filename}.tar" /home/Alice

mv "${filename}.tar" /var/backups/

gzip /var/backups/"${filename}.tar"
~
~
~
~
```

4. (30 Points) Create a crontab file to keep the scheduled task running for 3 minutes, then check the contents in the /var/backups directory. Your output should be look similar to the following:

```
File  Actions  Edit  View  Help

# Edit this file to introduce tasks to be run by cron.
# DO NOT EDIT THIS FILE - edit the master and reinstall.
# Each task to run has to be defined through a single line18:28 2020)
# indicating with different fields when the task will be run (via %p %s)
# and what command to run for the task be run by cron.
#
# To define the time you can provide concrete values for:
# minute (m), hour (h), day of month (dom), month (mon), run
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
# and day of week (dow) or use '*' in these fields (for 'any').
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
# daemon's notion of time and timezones.
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with: (including errors) is sent through
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/ (unless redirected).
#
# For more information see the manual pages of crontab(5) and cron(8)
# at 5 a.m every week with:
# m h dom mon dow  command
*/3 * * * * /home/backup.sh
~
```

5.

```
—(jpp@kali)-[~]
—$ ls /var/backups
alternatives.tar.0      dpkg.statoverride.1.gz
apt.extended_states.0  dpkg.status.0
pkg.arch.0             dpkg.status.1.gz
pkg.arch.1.gz          jpeck-03.28.2023-18.28.36.tar
pkg.diversions.0       jpeck-2023.03.28.18.31.04-18.31.07.tar
pkg.diversions.1.gz    jpp-2023.03.28-18.30.30.tar
pkg.statoverride.0
```

6. (10 Points) **Cancel** the crontab jobs.

```
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h dom mon dow   command
*/3 * * * * /home/backup.sh
```

7.

TASK B: SYSTEM CLEANUP (**EXTRA CREDIT**) [20 Points]

Scenario: In the above scenario, your system disk will be filled up eventually without cleaning up the old backups. Therefore, in this optional task, create a script that checks the number of backups you created in Task A. If the number of the backup file is more than a pre-defined threshold, the script will delete the old archives to maintain the backups under a reasonable size.

This script should do the following:

1. Count the number of backups created in Task A and determine if this number is larger than 3. Nope
2. Nothing should happen if the number of backups is less than the threshold, 3.
3. If more backup archives are detected, calculate the number of backups to delete. Then delete the old archives.

```
#!/bin/bash

# Define the directory where the backups are stored
BACKUP_DIR=/var/backups

# Define the threshold for the number of backups to keep
THRESHOLD=3

# Count the number of backup files in the directory
BACKUP_COUNT=$(ls $BACKUP_DIR/*.tar.gz | wc -l)

# Check if the number of backup files is greater than the threshold
if [ $BACKUP_COUNT -gt $THRESHOLD ]; then
    # Calculate the number of backups to delete
    DELETE_COUNT=$((BACKUP_COUNT - THRESHOLD))

    # List the backups in reverse chronological order (newest first)
    BACKUP_LIST=$(ls -t $BACKUP_DIR/*.tar.gz)

    # Loop through the list of backups and delete the old ones
    for BACKUP_FILE in $BACKUP_LIST; do
        if [ $DELETE_COUNT -gt 0 ]; then
            rm $BACKUP_FILE
            DELETE_COUNT=$((DELETE_COUNT - 1))
            echo "Deleted old backup: $BACKUP_FILE"
        else
            break
        fi
    done
fi
```

This is the script that I ran.

Note: As the script needs to write contents in the “/var/backups” folder, which is owned by root, you should consider the permission issue properly. (Using **sudo** to create crontab file)

Reference: How to Format Date for Display or Use In a Shell Script-

<https://www.cyberciti.biz/faq/linuxhttps://www.cyberciti.biz/faq/linux-unix-formatting-dates-for-display/unix-formatting-dates-for-display/>

Reference: How to append date timestamp to filename- <https://crunchify.com/shell-script-appendhttps://crunchify.com/shell-script-append-timestamp-to-file-name/timestamp-to-file-name/>

<https://crunchify.com/shell-script-append-timestamp-to-file-name/timestamp-to-file-name/>

