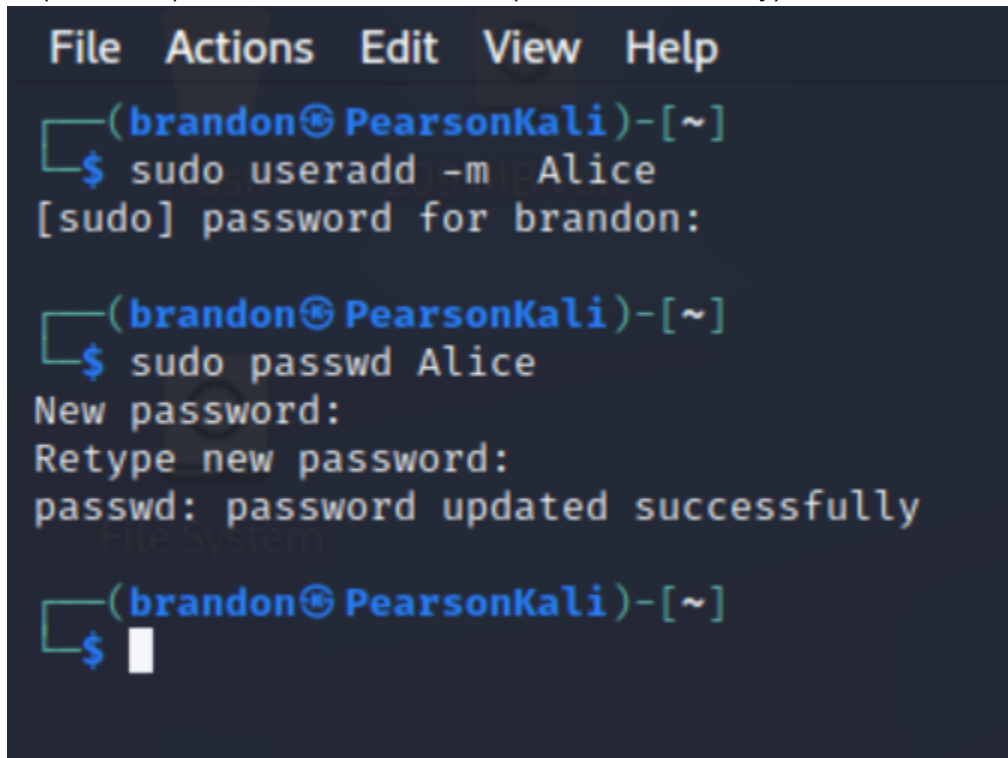


Task A - Backup your system (Using crontab) [100 points]

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 with a dark background and light-colored text. The window title is '(brandon@PearsonKali)-[~]'. The prompt is '\$'. The user enters 'sudo useradd -m Alice'. The prompt changes to '[sudo] password for brandon:'. The user enters a password (indicated by a white box). The prompt returns to '\$'. The user enters 'sudo passwd Alice'. The prompt changes to 'New password:'. The user enters a password (indicated by a white box). The prompt changes to 'Retype new password:'. The user enters the same password (indicated by a white box). The prompt changes to 'passwd: password updated successfully'. The prompt returns to '\$'. The user enters a command (indicated by a white box).

```
File Actions Edit View Help

(brandon@PearsonKali)-[~]
$ sudo useradd -m Alice
[sudo] password for brandon:

(brandon@PearsonKali)-[~]
$ sudo passwd Alice
New password:
Retype new password:
passwd: password updated successfully

(brandon@PearsonKali)-[~]
$
```

2. (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).

```
#!/bin/bash
midas=bpear
current_date=$(date +%Y.%m.%d.-%H.%M.%S)
tar_file=$midas-$current_date

echo $tar_file

(brandon@PearsonKali)-[~]
$ vi script9.sh

(brandon@PearsonKali)-[~]
$ chmod a+x script9.sh

(brandon@PearsonKali)-[~]
$ ls -l script9.sh
-rwxr-xr-x 1 brandon brandon 113 Mar 28 19:36 script9.sh
```

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.

```
#!/bin/bash
midas=bpear
current_date=$(date +%Y.%m.%d.-%H.%M.%S)
tar_file='/var/backups/' $midas-$current_date
gzip /var/backups/"$tar_file.tar"

echo $tar_file
```

3. (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
GNU nano 6.4 /tmp/crontab.Qmebre/crontab
# 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
*/1 * * * * /home/brandon/script9.sh >> /var/backups/

[ Read 25 lines ]
^G Help      ^O Write Out  ^W Where Is   ^K Cut        [ Execute    ^L Location   M-U Undo     M-A Set Mark  M-] To Bracket
^X Exit      ^R Read File  ^N Replace    ^P Paste      ^T Justify   ^G Go To Line M-E Redo     M-G Copy      ^_ Where Was

(brandon@PearsonKali)-[~]
$ ls /var/backups/
apt.extended_states.0  apt.extended_states.1.gz

(brandon@PearsonKali)-[~]
$ ls /var/backups/
apt.extended_states.0  apt.extended_states.1.gz  testoutput

(brandon@PearsonKali)-[~]
$ cat /var/backups/testoutput
bpbear-2023.03.28.-19.52.01

(brandon@PearsonKali)-[~]
$ cat /var/backups/testoutput
bpbear-2023.03.28.-19.52.01
bpbear-2023.03.28.-19.53.01

(brandon@PearsonKali)-[~]
$ cat /var/backups/testoutput
bpbear-2023.03.28.-19.52.01
bpbear-2023.03.28.-19.53.01
bpbear-2023.03.28.-19.54.01
```

#### 4. (10 Points) Cancel the crontab jobs

```
(brandon@PearsonKali)-[~]  
$ cat /var/backups/testoutput  
bpear-2023.03.28.-19.52.01  
bpear-2023.03.28.-19.53.01  
bpear-2023.03.28.-19.54.01  
bpear-2023.03.28.-19.55.01  
bpear-2023.03.28.-19.56.01  
bpear-2023.03.28.-19.57.01
```

```
(brandon@PearsonKali)-[~]  
$ sudo crontab -r
```

```
(brandon@PearsonKali)-[~]  
$ cat /var/backups/testoutput  
bpear-2023.03.28.-19.52.01  
bpear-2023.03.28.-19.53.01  
bpear-2023.03.28.-19.54.01  
bpear-2023.03.28.-19.55.01  
bpear-2023.03.28.-19.56.01  
bpear-2023.03.28.-19.57.01
```

```
(brandon@PearsonKali)-[~]  
$ sudo crontab -l  
no crontab for root
```