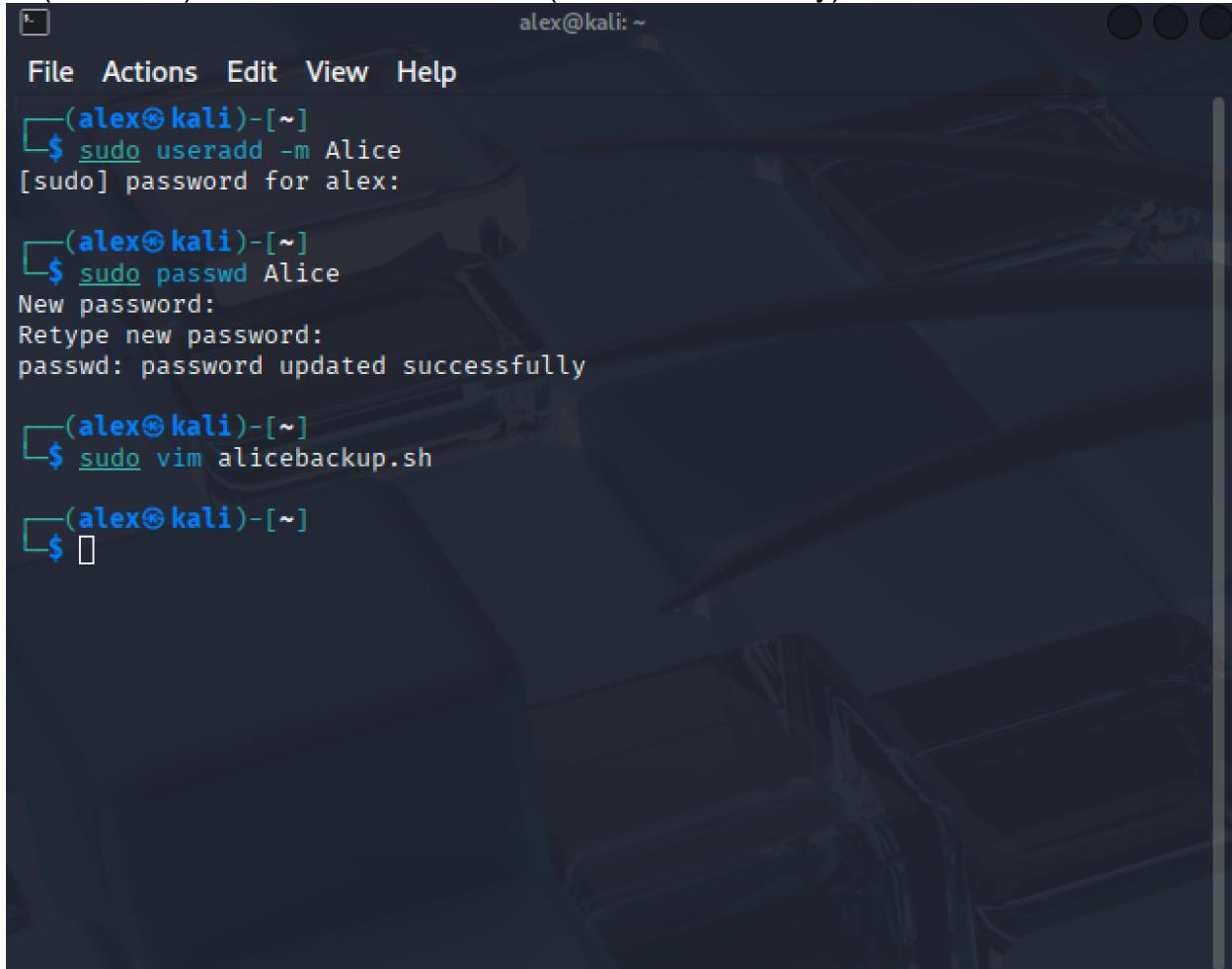


CYSE 270: Linux System for Cybersecurity
Assignment-9

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).



The screenshot shows a terminal window with a dark background and a light blue header bar. The header bar displays the text "alex@kali: ~". The terminal window contains the following text:

```
File Actions Edit View Help
└─(alex㉿kali)-[~]
└─$ sudo useradd -m Alice
[sudo] password for alex:
└─(alex㉿kali)-[~]
└─$ sudo passwd Alice
New password:
Retype new password:
passwd: password updated successfully

└─(alex㉿kali)-[~]
└─$ sudo vim alicebackup.sh

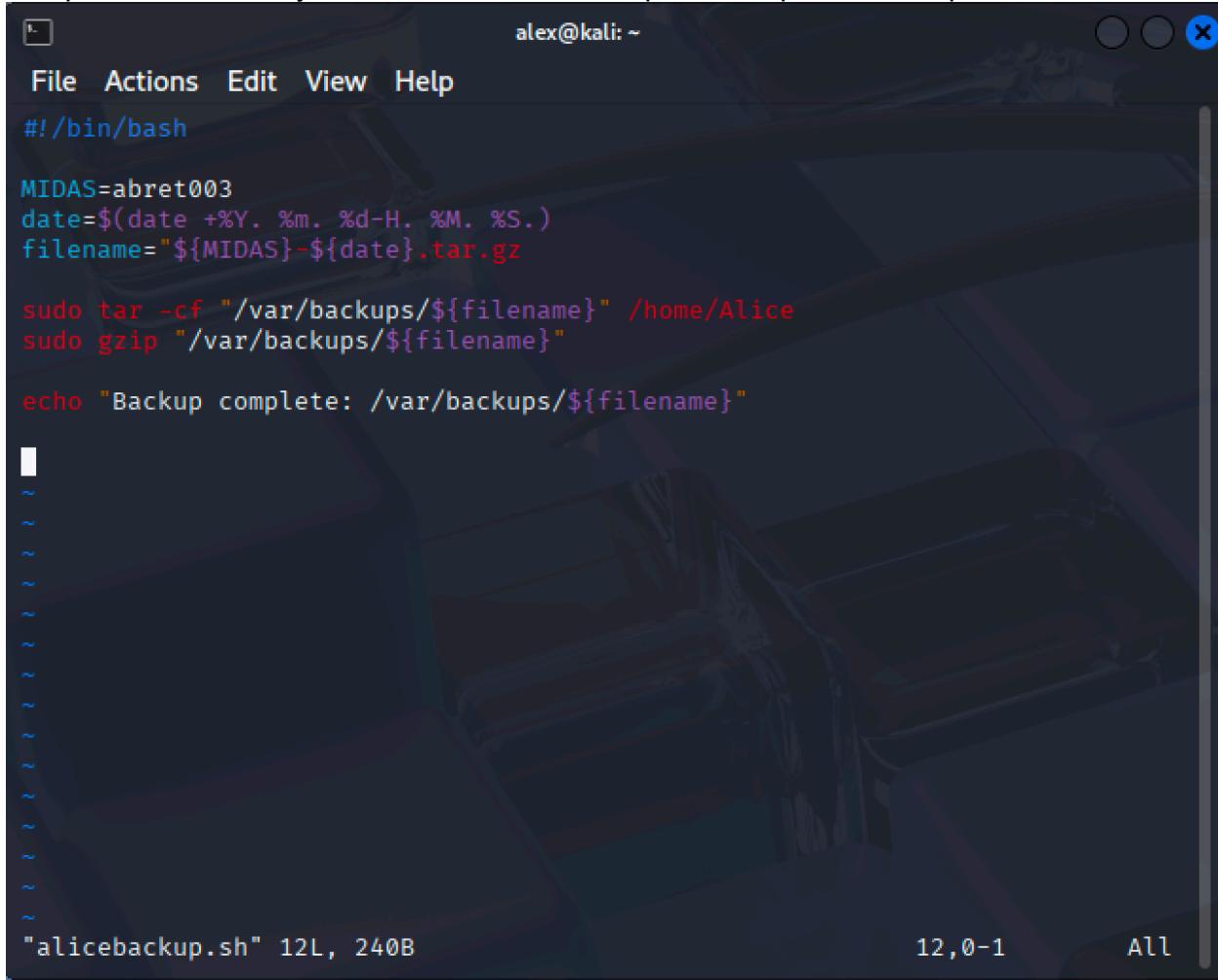
└─(alex㉿kali)-[~]
└─$
```

2. (50 Points) Write a shell script that backups Alice's home directory by creating a tar file (tape archive), using the following steps:

a. Do the following:

- Take 2 inputs with their values- your MIDAS name and current date (for example, midas=Mohammed).
- Create a variable named as filename that should be assigned the value as MIDAS-date (example output after executing the script would be like, Mohammed-2024.11.04-22.08.01.tar.gz).
- 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.



A screenshot of a terminal window titled 'alex@kali: ~'. The window contains a bash script named 'alicebackup.sh' with the following content:

```
#!/bin/bash

MIDAS=abret003
date=$(date +%Y. %m. %d-H. %M. %S.)
filename="${MIDAS}-${date}.tar.gz"

sudo tar -cf "/var/backups/${filename}" /home/Alice
sudo gzip "/var/backups/${filename}"

echo "Backup complete: /var/backups/${filename}"
```

The terminal window shows the script being typed in. The status bar at the bottom indicates the file is 12L, 240B, and the cursor is at position 12,0-1. The 'All' tab is selected.

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:

```
alex@kali: ~
File Actions Edit View Help
# 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/alex/alicebackup.sh >> /var/backups/alice.log
~
~
"/tmp/crontab.GyRkVE/crontab" 25L, 953B           1,1          All
```

4. (10 Points) Cancel the crontab jobs.

```
[(alex@kali)-[~]
$ sudo crontab -r
[sudo] password for alex:
```

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.
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.

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/linux-unix-formatting-dates-for-display/>

Reference: How to append date timestamp to filename:

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