Task Automation

Introduction and/or Background
Network and System Administrators often use automation to conduct time-consuming, routine, or repeatable tasks. This frees them up to be more productive in other parts of the job. In Linux, cron jobs and the at command are used to automate this type of day-to-day work.

The command crontab is the tool used manage all cron jobs. There are two sets of crontab files. There is the crontab file available to the root user. Anything scheduled from this account applies system wide. Then there is the crontab file that the individual account owners may use to schedule their own jobs as they see fit. A user crontab file only affect files and jobs that the user has ownership of. A user cronjob cannot override a root user cronjob.

Objectives
In this project/lab the student will:

- Use a text editor to create and execute cron jobs
- Schedule commands to be executed at a specific time using the at command

Equipment/Supplies Needed

- As specified in Lab 0.0.1.

Procedure
Perform the steps in this lab in the order they are presented to you. Answer all questions and record the requested information. Use the Linux Virtual Machine to perform lab activities as directed. Unless otherwise stated, all tasks done as a non-root user. If root access is needed use the sudo command.

Assignment
Lab Submissions Proof: Provide screenshots as indicated in the lab; upload your proof to Canvas for grading.

User Cron Jobs

We use **crontab -e** to load cron jobs into cron

Use **crontab -l** to list any jobs that may be currently loaded in crontab.

Example, create a job that runs in about 2-5 minutes from the current system time as reported by the **date** command. The job will create a file in **/home/<<user>>/CronTabIpaddr**, the contents will be the results of **/sbin/ifconfig**. Note: <<user>> is your current logged in account; eg. User1.

Execute the **date** command to get the system's current time.

1. **date**

Example:

Thu Mar 21 16:43:54 CDT 2019

Enter

2. **crontab -e**

at the prompt

If instructed to select an editor, press Enter to select option 1 (**/bin/nano**)

```
Select an editor.  To change later, run 'select-editor'.
  1. /bin/nano        <---- easiest
  2. /usr/bin/vim.tiny

Choose 1-2 [1]:
```

Example:

Add the following to the crontab file:

45 18 * * * /bin/ip addr > /home/<<user>>/CronTabIpaddr

Note change the items highlighted in green to match the results of the date command.

Note: In most cases the crontab points to a script, by pointing to a script one can change what the crontab does without having to edit the crontab file. In addition, it allows for more complex operations to be done.

Press Control X to quit the file, then Y to save changes.

Check to see if it was loaded by typing

3. **crontab -l**.

It should display something like this:

<Header Text>

45 18 * * * ip addr > /home/tsantos/CronTabIpaddr

After the 2-3 minutes have passed, check **/home/<<user>>**, you should see the file **CronTabIpaddr**, with a date time stamp, the same as you entered into the cronjob (see example below):

4. ls -l CronTabIpaddr

   Example:

   -rw-r--r-- 1 tsantos users 917 Mar 21 16:45 CronTabIpaddr

Record a screenshot of the **crontab -l** command, and the **ls -l /home/<<user>>/CronTabIpaddr** file.

Scheduling with at Command

Install the **at** command-line utility:

5. **su - <enter>** At the **#** prompt enter

6. **apt install at -y <enter>**

Verify **at** is running with the

7. **systemctl status atd.service**

status command (must be root). If the service is not running, start it with

8. **systemctl start atd.service**

Display the current time by entering the

9. **date**

command.

Find out who is logged on 2 minutes from now (i.e: hh:mm = 2 minutes from now).

Enter the following command:

10. At hh:mm

where <<hh>> hour and <<mm + 2>> minutes from the date command

11. who >> /home/<<user>>/AtSample

Exit the at editor with ctrl +d

View the scheduled jobs with the

12. **atq**

command.

Check the **/var/log/syslog** after 2 minutes to see if the date information is entered (use the **tail** command in a separate terminal session). Date information should be listed at the bottom of the log.

Schedule the same job to run tomorrow at noon

13. at noon tomorrow

14. who >> /home/<<user>>/AtSample

Schedule the date to be logged tomorrow at 2 PM

15. at 14:00 tomorrow

16. who >> /home/<<user>>/AtSample

View the scheduled jobs again. Record a screenshot showing created jobs.

Remove the jobs with the

17. atrm job_number

command.

Confirm that they are gone by viewing the scheduled jobs again. Record a screenshot showing removed jobs.

Set date/time Cron Jobs will run

Review section 5 for crontab in the man pages.

Review each of the crontab time / date entries below, indicate the time / date the crontab would run (see first example below)

30 4 1,15 * 5 – runs, 4:30am on the 1st and 15th of each month, and every friday

30 08 12 06 * _____

0 */2 * * * _____

30 10-18 * * * _____

0 8-17/1 * * _____

0 23 31 12 * _____

Rubric

Checklist/Single Point Mastery

Concerns Working Towards Proficiency	Criteria Standards for This Competency	Accomplished Evidence of Mastering Competency
	Criteria #1: Recorded screenshot of crontab -l command (10 points)	
	Criteria #2: Recorded screenshot of ls -l /home/<<user>>/CronTablpaddr command (10 points)	
	Criteria #3: Recorded screenshot of created scheduled jobs (10 points)	
	Criteria #4: Recorded screenshot of removed scheduled jobs (10 points)	
	Criteria #5: Recorded answer to question 2a (10 points)	
	Criteria #6: Recorded answer to question 2b (10 points)	
	Criteria #7: Recorded answer to question 2c (10 points)	
	Criteria #8: Recorded answer to question 2d (10 points)	

	Criteria #9: Recorded answer to question 2e (10 points)	
	Criteria #10: Recorded answer to question 2f (10 points)	