

**GETTING STARTED
WITH
SCHEDULING_JOBS**

➤ SCHEDULING JOBS:

- Scheduling jobs also known as “Automating System Tasks”.
- **Tasks**, also known as **jobs**, can be configured to run automatically within a specified period.
- Red Hat Linux is pre-configured to run important system tasks to keep the system updated.
- Task utilities are:
 - at
 - batch
 - cron and anacron

❖ AT JOBS:

- To schedule a **one-time task**, also called a job, to run once at a specific time.

SYNTAX: #at [options]

OPTIONS:

-l : Lists all jobs in the queue
-d : Removes job from the queue
-c : Job Description

→ To execute the job at 15:00, run:

#at 15:00

→ To execute the job on August 20 2030, run:

#at August 20 2030 (or) #at 082030

→ To execute the job 5 days from now, run:

#at now + 5 days

→ To run a script at 3pm:

#at 15:00

at> **sh /opt/my-script.sh** [Press Ctrl+D to save]

→ To view the list of pending jobs:

#atq (or) #at -l

→ To get a job description:

#at -c jobid

→ To delete a scheduled job:

#at -d jobid (or) #atrm jobid

#atq

❖ **BATCH JOBS:**

- To schedule a one-time task, also called a job, to run when the system loads average drops below the specified value (0.8).
- This can be useful for performing resource-demanding tasks or for preventing the system from being idle.
- The prerequisites for batch jobs are the same as for at jobs.

SYNTAX: \$batch

NOTE: **Batch** does not accept any parameters.

→ At the displayed at> prompt, enter the command to execute and press Enter:

#batch

at> **sh /opt/my-script.sh**

CONTROLLING ACCESS TO AT & BATCH:

- You can restrict access to the **at** and **batch** commands for specific users.
- To do this, put user names into **/etc/at.allow** or **/etc/at.deny** according to these rules:
 - If the **at.allow** file exists, only users listed in the file are allowed, and the **at.deny** file is ignored.
 - If **at.allow** does not exist, users listed in **at.deny** are not allowed.

❖ CRON & ANACRON:

- These are daemons that can schedule execution of **Recurring tasks** to a certain point in time.
- A cron job is only executed if the system is running on the scheduled time.
- If the system is not running on at the time when a job is scheduled, the job is not executed.
- **Anacron** remembers the scheduled jobs if the system is not running at the time when the job is scheduled. The job is then executed as soon as the system is up.

CRONTAB FORMAT:

- **Minute** : From 0 to 59
- **Hour** : From 0 to 23
- **Day** : From 1 to 31
- **Month** : From 1 to 12
- **Day of week** : From 0 to 7, where 0 or 7 represents Sunday
- **Username** : Specifies the user under which the jobs are run
- **Command** : The command to be executed

SYNTAX:

#crontab [options] [-u user]

- e** : Edit the user crontab
- l** : Lists the user crontab
- r** : Deletes the user crontab
- i** : Prompts before deleting user's job

PREREQUISITES FOR CRON JOBS:

→ Install the cronic package:

#yum install cronic -y

→ The crond service is enabled at boot time:

#systemctl enable crond.service

→ Start the crond service for the current session:

#systemctl start crond.service

#systemctl status crond

CRONTAB EXAMPLES:

→ To open a crontab editor for current user:

#crontab -e						
0	10	*	*	*	date	Run at 10:00am every day
5	12	*	*	*	ls /opt	Run at 12:15pm every day
0	18	*	*	MON-FRI	sh script.sh	Run at 6:00pm every Monday through Friday
0	8	1	*	*	cp file1 file2	Run at 8:00am every 1st day of the month
0/5	*	*	*	*	uptime	Run every 5 minutes

→ To list scheduled jobs:

#crontab -l

→ scheduling a job for specific user:

#crontab -u raju -e

To List scheduled jobs:

#crontab -u raju -l

CONTROLLING ACCESS TO CRON:

- To restrict the access to Cron, you can use files: /etc/cron.allow
/etc/cron.deny
- The /etc/cron.allow File:
 - If it exists, only these users are allowed (cron.deny is ignored).
 - If it doesn't exist, all users except cron.deny are permitted.
- The /etc/cron.deny File:
 - is empty, all users are allowed (Red Hat Default).
- For Both Files:
 - If neither file exists, root only.