Scheduling (Core)

Linux Commands Course · Section 15

Goal

Learn how to automate and schedule tasks in Linux using cron for recurring jobs and at for one-time jobs.

Automation is essential for backups, maintenance, and system monitoring.

What Is Job Scheduling?

Linux can run commands automatically at specific times or intervals.

Two main tools handle this:

- **cron** → recurring jobs (daily, hourly, weekly, etc.)
- at → one-time jobs

The scheduler runs in the background and executes tasks even if you're not logged in.

Recurring Jobs — cron

cron reads scheduled jobs from special files called crontabs. List current user's scheduled jobs: Edit your crontab: Each line defines one job using this format: * * * * * command to run Day of week (0-7) (Sunday = 0 or 7) Day of month (1-31) Hour (0-23) Minute (0-59) Example: run a script every day at 2:30 AM 30 2 * * * /home/student/backup.sh

Special Cron Keywords

You can use shortcuts instead of the 5-field format:

Keyword	Meaning		
@reboot @daily	once at startup once a day		
@hourly	every hour		
@weekly	once a week		
@monthly	once a month		

Example:

@reboot /usr/local/bin/monitor.sh
@daily /usr/local/bin/cleanup.sh

System-Wide Cron Directories

In addition to user crontabs, system-wide jobs live in these directories:

Location	Purpose
/etc/crontab /etc/cron.hourly/ /etc/cron.daily/ /etc/cron.weekly/ /etc/cron.monthly/	main system cron file scripts run every hour scripts run daily scripts run weekly scripts run monthly

System crontab includes an extra field for the **user** to run as:

m h dom mon dow user command
17 * * * * root run-parts /etc/cron.hourly

Controlling Cron Jobs

List cron service status (systemd-based systems):				
	systemctl status cron			
Restart it if needed:				
	sudo systemctl restart cron			
You can temporarily disable user cron jobs by commenting them out in crontab -e.				

Viewing Cron Logs

Cron logs are usually stored under /var/log.

sudo grep CRON /var/log/syslog

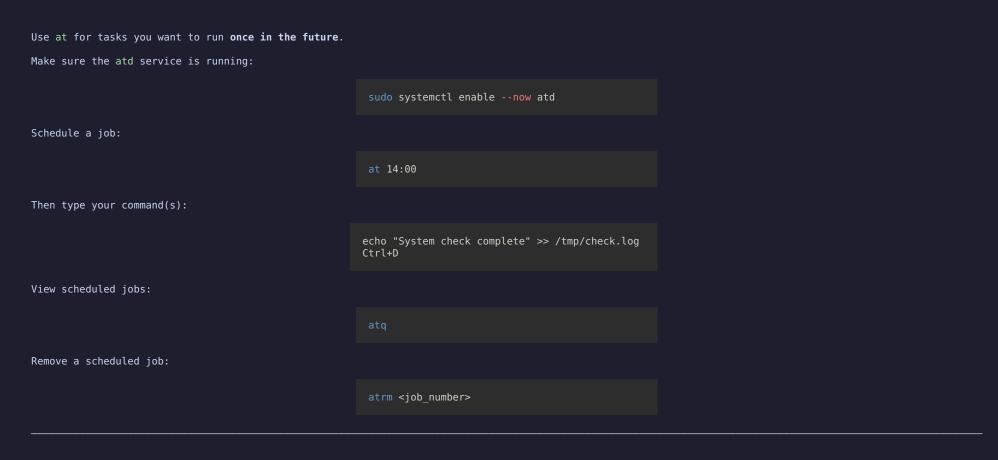
Or for Red Hat-based systems:

sudo grep CROND /var/log/cron

You can also redirect cron job output manually in your job definition:

0 1 * * * /usr/local/bin/backup.sh >> /var/log/backup.log 2>&1

One-Shot Jobs — at



Flexible Time Syntax with at

Examples of valid scheduling times:

```
at now + 1 hour
at midnight
at 8pm tomorrow
at 10:30am next Monday
```

at is perfect for one-off delayed commands or testing automation tasks.

Examples — Real Use Cases

```
Daily backup with cron:

0 2 * * * /usr/local/bin/backup.sh

Run maintenance 5 minutes from now with at:

echo "apt update && apt upgrade -y" | at now + 5 minutes

Weekly report via email:

0 9 * * 1 /usr/local/bin/report.sh | mail -s "Weekly Report" admin@example.com
```

Recap

- cron recurring tasks (crontab -e, /etc/cron.*)
 at one-time jobs (at, atq, atrm)
 systemctl status cron / atd ensure schedulers are active
 Use log redirection for auditing outputs

Automation keeps your system consistent, efficient, and hands-free.

Practice

- Schedule a command to run every minute (for testing).
 Add a daily cleanup job at midnight with crontab -e.
 View your current crontab with crontab -l.
 Schedule a one-time notification with at now + 2 minutes.
 Check logs to confirm that your jobs ran successfully.

Next Up

Bash Scripting (Core → Plus) — automating with logic, variables, and control flow.