Scheduling Future Tasks

Unit objectives

After completing this unit, you will learn:

- Scheduling a Deferred User Job
- Scheduling Recurring User Job
- Scheduling using anacrond
- Scheduling using Systemd Timer Unit
- Managing Temporary Files
- Cleaning Temporary Files Manually

Scheduling a Deferred User Job

- Scheduled commands called tasks or jobs
- Deferred – jobs that runs in the future
- Handled by **atd** daemon
- Scheduling tasks run once at specific time using **at, atq**
- Scheduling tasks that runs once system is not busy using **batch**

The at scheduler

- at TIMESPEC to schedule new job
- at command read from stdin
- Ctrl+D to save configuration / Ctrl+C to cancel and exit
- Use input redirection from a script

```
# at 5pm tomorrow < myscript
```

- TIMESPEC argument
  - now +5min
  - teatime tomorrow (16:00 / 4pm tomorrow)
  - noon +4 days (4 days later at 12pm)
  - 5pm dec 31 2021 (exactly at 5pm on 31-Dec-2021)

- Non-root user can only see and control their own jobs

More example – at scheduler

- At exact date and time

```
# at –v 10am jul 31
# at –v 17:45 dec 31
# at –v 17:45 dec 31 2022
# at –vt 203012311730
```

- Today

```
# at 5pm today
# at –v now +30 min
# at –v now +2 hour
```

- Tomorrow

```
# at –v tomorrow
# at 5pm tomorrow
```

- Noon

```
# at –v noon tomorrow
# at –v noon +2 week
```

- Midnight

```
# at –v midnight +2 year
# at –v midnight tomorrow
# at –v midnight today ;)
```

- Next few days
- # at –v teatime +10 day
- # at –v midnight next day

More example – at scheduler

- Noon

```
# at –v noon tomorrow
# at –v noon +2 week
```

- Midnight

```
# at –v midnight +2 year
# at –v midnight tomorrow
# at –v midnight today ;)
```

- Next few days
- # at –v teatime +10 day
- # at –v midnight next day

- Next week

```
# at –v next week
# at –v 12pm +4 week
```

- Next month

```
# at -v next month
# at –v +2 month
```

More example – batch scheduler

- At once after system not busy

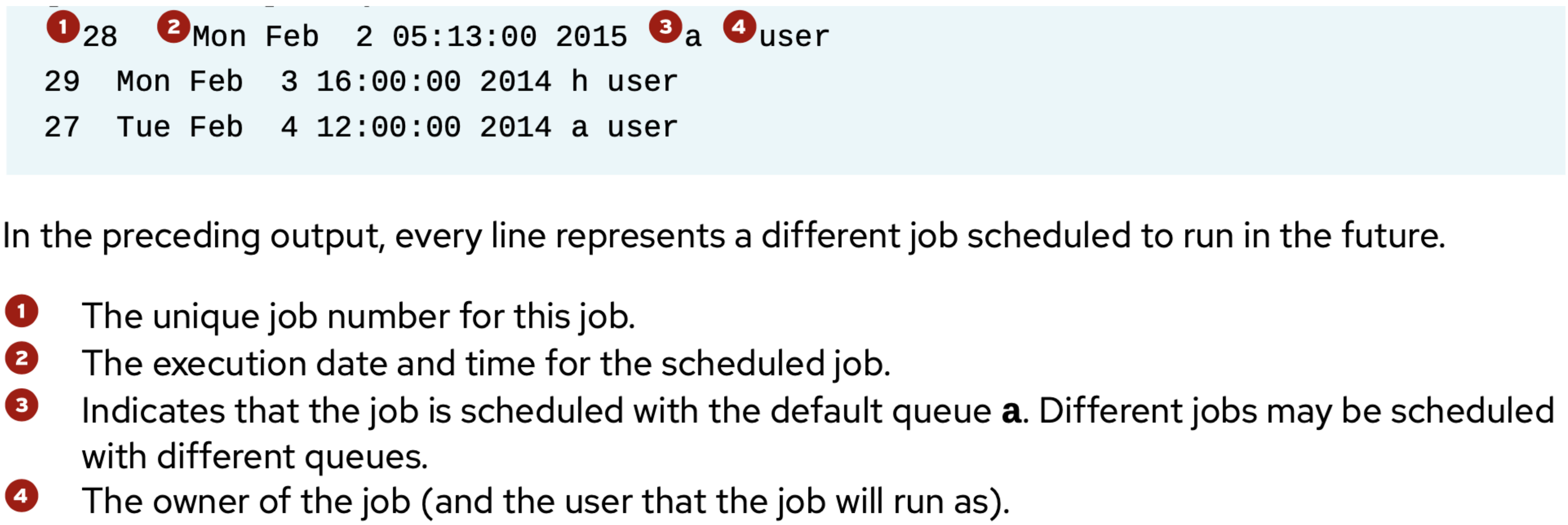
```
# batch
> dnf –y install httpd mariadb
> systemctl start httpd
> systemctl start mariadb

# at –b
> dnf –y install httpd mariadb
> systemctl start httpd
> systemctl start mariadb
```

List jobs

- List all jobs

# at –l or

# atq



```
❶28   ❷Mon Feb  2 05:13:00 2015 ❸a ❹user
29   Mon Feb  3 16:00:00 2014 h user
27   Tue Feb  4 12:00:00 2014 a user
```

In the preceding output, every line represents a different job scheduled to run in the future.

❶ The unique job number for this job.
❷ The execution date and time for the scheduled job.
❸ Indicates that the job is scheduled with the default queue **a**. Different jobs may be scheduled with different queues.
❹ The owner of the job (and the user that the job will run as).

- Display commands of specific job

# at –c 2

Using script and Queue

- Instead of using stdin, use –f script

```
# vi /tmp/msg
echo "Hello World" | wall
# chmod +x /tmp/msg
# at –v noon jan 1 –f /tmp/msg
```

- Queue value
  - ranging a-z, A-Z
  - a-z with higher letter run with less priority (higher nice value)
  - A-Z letter queue set priority for batch jobs

  ```
  # at –vq g noon tomorrow
  ```

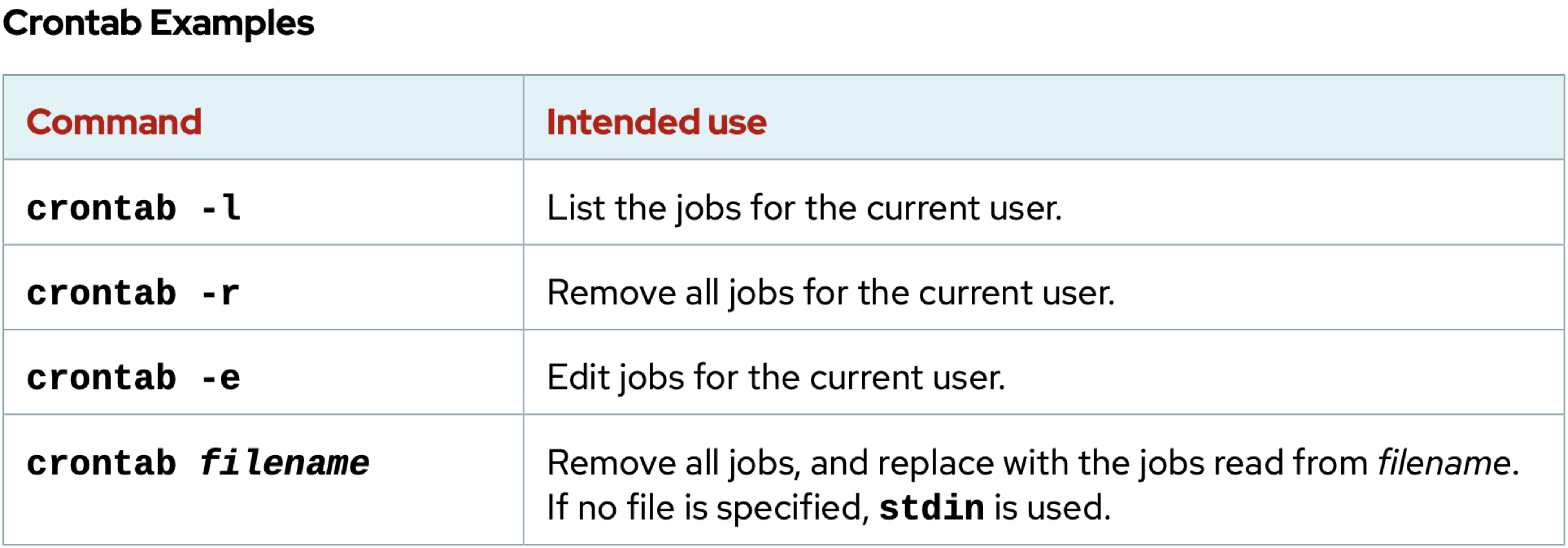
Remove jobs

- at -r <job id>
- atrm <job id>

- All users' jobs listed in /var/spool/at
  - only accesible by root

Scheduling Recurring User Job

- Run jobs repeatitively
- Handled by crond daemon
- Create job with **crontab** command
- If output not set with redirection, crond send via email to user



Crontab Examples

Command	Intended use
`crontab -l`	List the jobs for the current user.
`crontab -r`	Remove all jobs for the current user.
`crontab -e`	Edit jobs for the current user.
`crontab` ***`filename`***	Remove all jobs, and replace with the jobs read from *filename*. If no file is specified, **`stdin`** is used.

User Job Format

- EDITOR=vi; export EDITOR
- User EDITOR default to vim
- Fields in crontab file

Minutes Hours "Day of Month" Month "Day of Week" Command

- x-y for range
- x,y for lists
- * every

15 12 15 * Fri command

*/5 * * * 1 command

30 17 1-5 6,8,10 * command

*/10 9-16 * Jul * command

Scheduling Recurring System Jobs

- Best to run administrative commands using system accounts
- Instead of crontab command, use system-wide crontab files
  - /etc/crontab
  - /etc/cron.d/
- Or simply place executable script file into
  - /etc/cron.hourly
  - /etc/cron.daily
  - /etc/cron.weekly
  - cron.monthly

The anacrontab

- Ensure jobs always run
  - Do not get skipped even system turned off / reboot
- /etc/anacrontab
  - Send job's output to root's mail
  - run-parts scripts in /etc/cron.daily, /etc/cron.weekly, /etc/cron.monthly
- period in days field
  - interval in days for job to repeat
  - accept integer or macro (@daily, @weekly, @monthly)
- Delay in minutes field
  - Amount of time crond daemon should wait before start job
- Job Identifier
  - To identify the job in log messages
- START_HOUR_RANGE:3-22
  - Allow anacrond to run jobs between 3am to 10pm only

Systemd Timer

- New scheduling function RHELv7 onward
- Handled by systemd
- Used as part / dependency check with services
- Logs into journals for easier debugging purposes
- List timer unit in systemd

```
# systemctl -at timer
UNIT                      LOAD   ACTIVE  SUB     DESCRIPTION
dnf-makecache.timer          loaded active waiting dnf makecache --timer
systemd-tmpfiles-clean.timer loaded active waiting Daily Cleanup of Temporary
Directories
unbound-anchor.timer         loaded active waiting daily update of the root trust anchor for
DNSSEC
```

Sample Timer Unit

- All timer units
- # ls /usr/lib/systemd/system/*.timer
- cat mdcheck_start.timer

```
...
[Timer]
OnCalendar=Sun *-*-1..7 1:00:00
...
```

- Above signifies this timer activates the corresponding unit every sunday but falls on 1-7th of every month and year at 1am

- Enable timer unit

```
# systemctl enable –now mdcheck_start.timer
```

- If you modify content in *.timer directly, reload daemon

```
# systemctl daemon-reload
```

More - Sample Timer Unit

- Runs every 10 minutes:
- OnCalendar=*:00/10

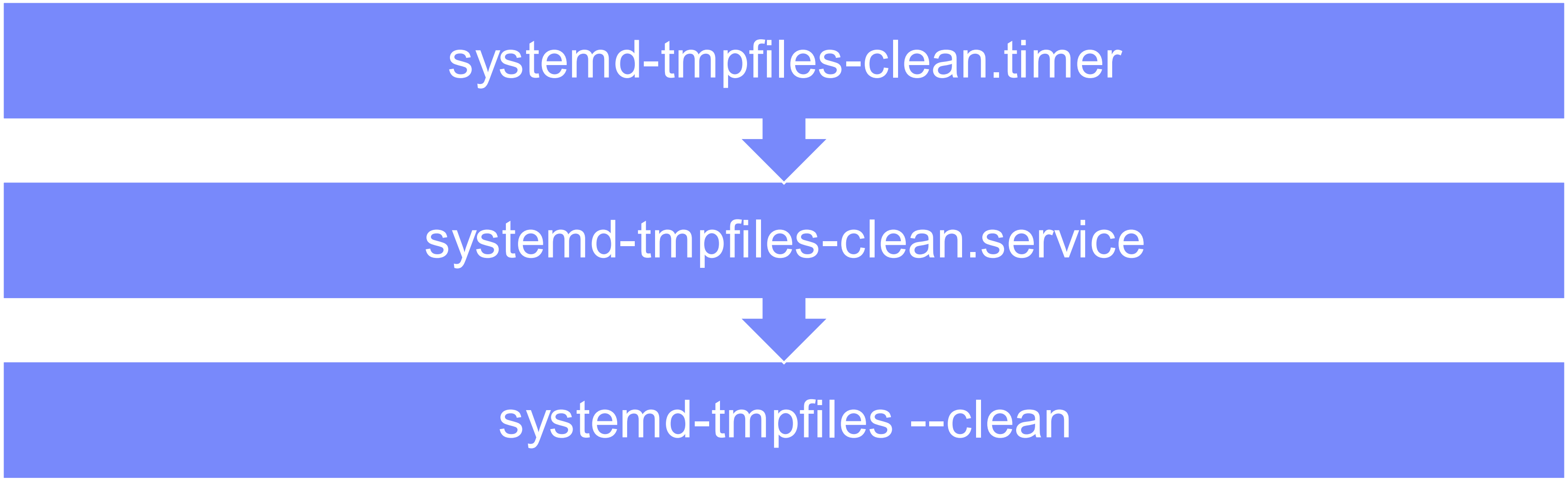
- Runs at specific date and time:
- OnCalendar=2025-01-01 12:00:05

- Runs at the 12:35:10, 12:37:10 and 12:39:10 times everyday in the month-year July-2025:
- OnCalendar=2025-07-* 12:35,37,39:10

Managing Temporary Files

- Common to create large number of temp files
- /tmp : User temp data, temp files
- /run : system / daemon volatile temp data
- RHEL v7, systemd-tmpfiles
  - provides structured and configurable method
  - reads configuration in /usr/lib/tmpfiles.d/*.conf, /run/tmpfiles.d/*.conf, /etc/tmpfiles.d/*.conf
  - Delete files/directories
  - Create files/directories
  - Change permission of files/directories

Cleaning temp files with systemd timer



systemd-tmpfiles-clean.timer

systemd-tmpfiles-clean.service

systemd-tmpfiles --clean

The systemd-tmpfiles location

1. /etc/tmpfiles.d/*.conf
   - configured by administrators
   - override vendor-provided
2. /run/tmpfiles.d/*.conf
   - volatile files
   - configured by daemons/applications to manage their own runtime temp files
3. /usr/lib/tmpfiles.d/*.conf
   - provided by relevant RPM package.
   - Should not edit these files

Cleaning Temporary Files Manually

- systemd-tmpfile --clean
  - parses same configuration files as systemd-tmpfile –create
  - but this will purge all files which have not accessed, changed, modified more recently
  - “maximum age” are defined in configuration files
- Format

```
d /run/systemd/seats 0755 root root –
```

- Create above directory if it does not exists with respective ownership and permission.

```
r /var/lib/rpm/__db.*
```

- with --clean option, delete all file start with __db in /var/lib/rpm

Cleaning Temporary Files Manually – More example

```
D /home/student 0700 student student 1d
```

- Create /home/student directory if it does not exists. If it does exists, empty it of all contents. When systemd-tmpfiles ---clean is run, remove all files which have not been accessed, changed or modified in more than one day

```
L /run/fstablink – root root - /etc/fstab
```

- Create symbolic link /run/fstablink to /etc/fstab. Never automatically purge this line

Checkpoint

1. Which command displays all user jobs that are currently scheduled to run as deferred jobs?
   a) atq
   b) atrm
   c) at –c
   d) at --display

2. Which commands removes deferred user job that has job id 5? [choose all that applied]
   a) at -q 5
   b) at -r 5
   c) at -c 5
   d) atrm 5

3. Which command displays recurring user job scheduled for currently logged-in user?
   a) crontab -r
   b) crontab -l
   c) crontab -u
   d) crontab –v

4. True or False: atq is the daemon that handles all deferred jobs

Guided Exercise

Topic	Page number on student-guide.pdf	Time (min)
Schedule a Deferred User Job	38	10
Schedule Recurring User Jobs	44	10
Schedule Recurring System Jobs	50	10
Manage Temporary Files	56	10

Unit summary

Having completed this unit, you should be able to:

- Schedule a Deferred User Job
- Schedule Recurring User Job
- Schedule using anacrond
- Schedule using Systemd Timer Unit
- Manage Temporary Files
- Clean Temporary Files Manually