Scheduling tasks using at command in Linux - kifarunix.com

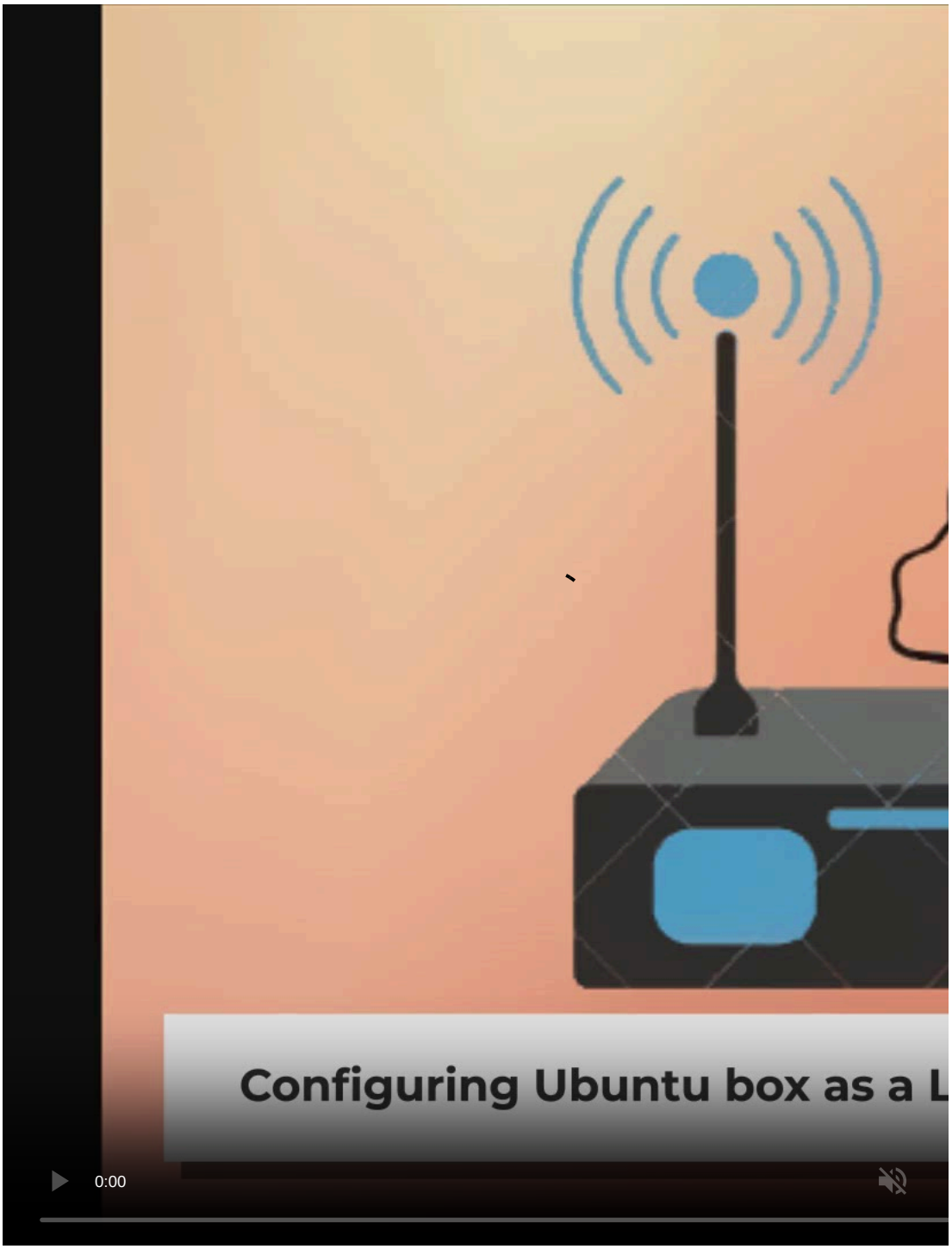
Published: 2019-09-07 · Archived: 2026-04-06 03:10:48 UTC

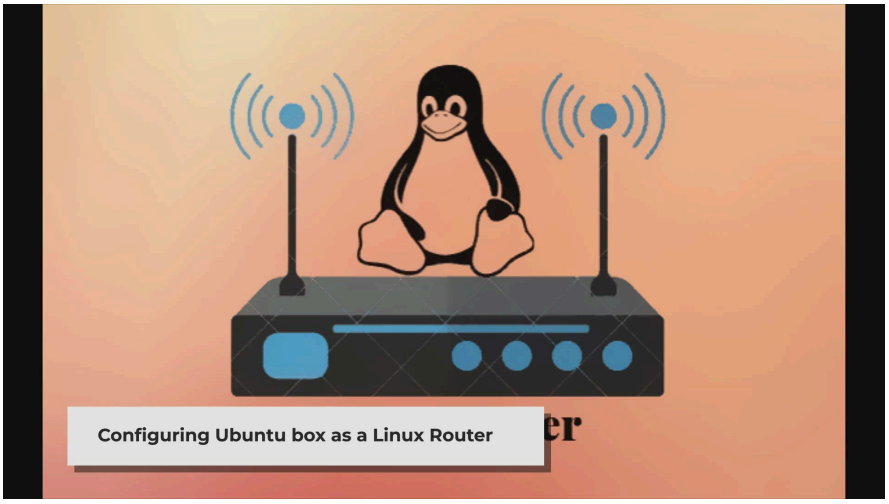
Our previous two guides, whose links are below, we covered how to schedule system tasks or jobs in Linux using the `cron` and `anacron` commands. This guide will focus on scheduling tasks using `at` command in Linux systems.

How to Schedule Cron Jobs/Tasks in Linux/Unix

Scheduling tasks using anacron in Linux/Unix

1. x





Video Player is loading.

2. 1. 

Now Playing

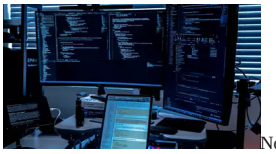
   Up Next*How to Configure Ubuntu 20.04 as a Linux Router: A Step-by-Step Guide*

   3:03

2. 
Now Playing

   Up Next*How To Use Task Manager In Linux Mint*

   2:03

3. 
Now Playing

   Up Next*Windows Users Try Linux for a Week: Surprising Results and New Perspectives*

   4:18

4. 

Now Playing

   Up Next*Schedule a Cron Job task in Linux - Ansible module cron*

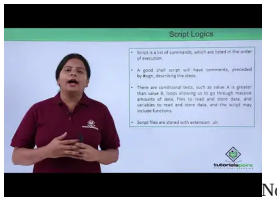
   11:12

5. 

Now Playing

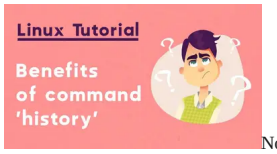
   Up Next*How to Reset Task Manager to default in Windows 11*

   2:38

6. 

Now Playing

   Up Next*Linux - Basic shell scripting*

   7:09
7. 

Now Playing

   Up Next*How to use history command for better CLI experience | Linux tutorial*

   5:14
8. 

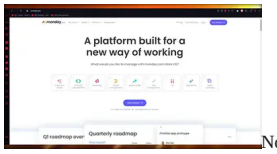
Now Playing

   Up Next*Wiring and Testing Our: P1-M622-16DR Practical Tutorial!*

   12:24
9. 

Now Playing

   Up Next*5 Essential Linux Skills: From Rescue Disks to Boot Fixes*

   2:34
10. 

Now Playing

    Up Next*How To Use Monday.Com For Task Management in 2024*

    8:17

x

`at` utility is used to schedule a one-time task. These are the tasks that are executed only once at a specific time.
`at` reads commands from standard input or from the file that is specified using the `-f` option.
Jobs scheduled using `at` are executed by the `atd` service.

Scheduling tasks using at command

The at command ordinarily expects time as the argument.

```
at TIME
```

Where TIME can be specified using keywords such as;

- `noon`, `midnight` or `teatime (4:00 PM)`,
- `next week`, `tomorrow`, `teatime tomorrow`, `next tuesday`, `next Tue`, `next Tuesday`, `next month`.

When executed with no commands or scripts passed to it, `at` commands runs interactively and expects you to enter the commands to execute from the prompt.

```
 at 11 PM Dec 2
at>
```

At the prompt, enter the command to run. For example;

```
 at 11 PM Dec 2
at> echo "This is a test at Job > /tmp/test-at-job"
```

```
 at 11 PM Dec 2
at> df -hT
```

When done typing commands, press Ctrl+D to terminate input. When you press Ctrl+D, you will see `<EOT>` at the command line.

```
 at 11 PM Dec 2
at> df -hT
```

You can as well pass the commands to execute to at command from command line by specifying the file with commands using the `-f` option. For example, consider a file called `mycommands.sh` with two shell scripts

```
 cat mycommands.sh
/home/me/myscripts/clean-tmp.sh
/home/me/myscripts/backup.sh
```

To have at execute these commands next week the same day at midnight;

```
at -f mycommands.sh midnight next week
```

You can as well use the standard input redirection symbol;

```
at teatime next month < mycommands.sh
```

You can also pipe tasks/commands to `at` command. For example to create a notification using the `notify-send` command,

```
echo 'notify-send "You need to refill the gas"' -t 1000 | at 17:00
```

This will send you a notification at 5 PM to refill the gas.

NOTE:

- For all the `at` examples that follows below, you can either pass the the commands or scripts or tasks/jobs to be executed to at command or you can interactively enter them.

To run the job one week from today, same time as when the at job was scheduled;

```
at next week
```

To run the job at 4 PM tomorrow

```
at teatime tomorrow
```

Run a job on next Tuesday, same time as when the job was scheduled.

```
at next tue
```

Run the job next month, same date at midnight.

```
at midnight next month
```

- using the keyword `now` plus a time period. For example, if to schedule a task to run 4 hours from now, use the time period `now + 4 hours`.

- For example;
  To run the job today at 4PM plus 2 hours, that is at 6PM

```
at 4PM + 2 hours
```

To run the job one and half hour from now;

```
at now + 90 minutes
```

To run the job two minutes from now;

```
at now + 2 minutes
```

To run a job next on monday 2 hours after the same time when the job was scheduled;

```
at monday + 2 hours
```

- using a time of the day either in `24-hour` or `12-hour` clock system. For example; `16:00` or `4:00 PM`.
- For example;
  To run the job at 5PM today;

```
at 5:00 PM or at 5 PM or at 17:00
```

- using `time of the day` and `date`.
  - The date can be specified in the format `DD.MM.YY`, `MMDDYY` or `MM/DD/YY`.
  - `MONTH Date, Year` or `MONTH Date Year` or `MONTH Date`. You can write the first three letters of the month e.g `Jan` or `jan`.
  - For example,
    To run the job on first october, year 2019 same time as when the job was scheduled

```
at 12:30 PM 10/01/19 or at 12:30 PM 01.10.19 or at 12:30 PM 100119
```

To run the job on 2nd Jan, 2020 same time as when the job was scheduled

```
at Jan 1, 2020 ot at Jan 1 2020
```

To run the job at 11 PM on 2nd Dec, this year;

```
at 11 PM Dec 2
```

List scheduled at jobs

At command has a utility called `atq` that can be used to list at jobs that are pending execution.
To simply list the at jobs, run;

```
atq
```

If there are any jobs pending execution, you will see them on the output;

```
atq
```

```
 31  Mon Dec  2 23:00:00 2019 a root
33  Sun Oct  6 16:00:00 2019 a root
32  Fri Sep 13 00:00:00 2019 a root
```

Similarly, you can use at command with `-l` option.

```
at -l
```

Delete Scheduled at jobs

Scheduled at jobs can be removed using the `atrm` utility or by passing option `-r` to at command.

```
atr -r <JOB ID>
```

or

```
atrm <JOB ID>
```

For example, based on the output of the `atq` command above, you can remove at job with job number 31 using the command;

```
at -r 31
```

or

```
atrm 31
```

To remove all the jobs in the queue, simply obtain the job numbers and remove them as follows;

```
atrm $(atq | cut -f1)
```

There are other options that are aliased to `atrm` command.

```
at -d <JOB ID>
```

Print Listed At Jobs to STDOUT

To print listed at jobs to standard output, simply use the command;

```
at -c <JOB ID>
```

For example, to cat the at job number 23 to standard output.

```
at -c 23
```

Controlling Access to At

It is also possible to control who can run at jobs using the `/etc/at.allow` and `/etc/at.deny` files.

- only users listed in the `at.allow` file are allowed to use `at`
- users listed in `at.deny` file are not allowed to use `at` .

Using batch command

Another command almost similar to at command is `batch` command. This command is used to schedule tasks that can only be executed when system load drops below 1.5 or any value specified in
Batch command does not accepts any parameters and runs interactively.

```
 batch
at>
```

For example to execute the script, `/home/me/myscripts/clean-tmp.sh` , simply run `batch` command and enter the script at the `at` prompt.
Press Ctrl+d once you are done typing commands;

```
 batch
at> /home/me/myscripts/clean-tmp.sh
at>
```

`batch` command is no longer maintained on most Linux/Unix distributions.

Read more about `at` command on `man at` .
