Schedule a controlled task with a systemd timer

A systemd timer separates a schedule from the service that performs the work. This makes status and logs easier to inspect, but does not make an unsafe script safe. These instructions target system-level systemd on Debian or Ubuntu with administrative access. Start with a harmless task and use the installed manual for options supported by your release.

Prepare a bounded service

Choose a task that is safe to run once and whose success can be verified. Do not begin with database deletion, payment collection or an untested backup restore. Record the required account, working directory and expected duration. If an existing cron or panel schedule performs the same job, keep track of it to prevent duplicate execution.

A one-shot service might look like this after you adapt it to a real, reviewed script and existing service account:

```
[Unit]
Description=Generate the daily application report

[Service]
Type=oneshot
User=example
WorkingDirectory=/srv/example
ExecStart=/usr/local/libexec/example-report
```

Save it as an appropriately named service under `/etc/systemd/system/`. Use absolute executable paths and keep credentials outside command arguments. Confirm the account can read its configuration and write only its intended output directory.

Define the schedule deliberately

A matching timer can use the following schedule. The example service and timer must share the same base name unless the timer explicitly names another unit.

```
[Unit]
Description=Schedule the application report

[Timer]
OnCalendar=*-*-* 03:15:00 UTC
Persistent=false

[Install]
WantedBy=timers.target
```

Use `systemd-analyze calendar` with the chosen expression to preview upcoming times. Check server time synchronization. Persistent catch-up behaviour deserves a decision: enabling it can run a missed calendar task after downtime, which is useful for some jobs and wrong for others. A timer is not a hard real-time scheduler.

Validate before enabling

Inspect both files with `systemd-analyze verify`, supplying their actual paths. Reload the manager after saving approved changes. Run the one-shot service manually only when its side effects are understood, then inspect its status, output and journal. A syntax-valid unit can still fail because its script or permissions are wrong.

Once the service works, enable and start only the timer using its exact name. Review `systemctl list-timers --all` and the next scheduled time. Starting the service directly runs the task; starting the timer schedules it. Keep those operations distinct.

Verify a scheduled run

Inspect both the timer and service status after the scheduled time. Read the service journal and check the actual report or result. Systemd generally will not start another instance of the same service while it is already active, but other scripts or schedules can still duplicate work. Disable the old scheduler only after confirming the replacement and understanding any overlap window.

Roll back cleanly

Stop and disable the specific timer if the schedule is wrong. This does not necessarily stop an already running service; assess that job before interrupting it. Restore the previous approved schedule if needed and preserve logs. Do not remove unrelated unit files or reset global systemd settings to recover one task.

Related instructions

Investigate systemd services. Check Linux time and NTP.

Reference: Official documentation. Check the documentation installed on your own operating-system release when an option differs.

Revision #1
Created 2026-11-09 10:42:43 UTC by CloudExa Hosting Admin
Updated 2026-11-09 10:42:43 UTC by CloudExa Hosting Admin