

Scheduling & Certificates

- [Deploy cron jobs to many servers from one script folder](#)
- [Reliable Let's Encrypt renewals on RHEL / Rocky](#)

Deploy cron jobs to many servers from one script folder

A lightweight way to manage scheduled jobs on a handful of Linux servers without Ansible. Keep one folder of scripts and cron files on an admin box, copy it to every server, and let each server install the cron files meant for it.

Folder layout

```
/opt/scripts/  
  backup_scripts/...  
  crontab_scripts/  
    cron0hourly      # all servers  
    cronloadmonitor  # all servers  
    web01_croncertbot # only on the host whose short name is "web01"  
    db01_cronbackupmysql # only on db01  
    profile.d/*.sh    # login scripts for every server
```

- **Naming rule:** files starting with `cron` go to every server. Files named `<shorthost>_<name>` go only to that host.
- Every file uses `/etc/cron.d` format, which includes the user field:

```
*/5 * * * * root /opt/scripts/loadmon.sh >/dev/null 2>&1
```

- Put shell logic in scripts, never in a cron file. Cronie skips the **whole file** if one line isn't a valid cron entry.

The installer (runs on each server)

Install only files with your own prefix, and remove only your own stale files. Don't wipe `/etc/cron.d`, because packages put their own jobs there (`0hourly`, `raid-check`, monitoring agents).

```
#!/bin/bash
set -euo pipefail
[[ $EUID -eq 0 ]] || exit 1
SRC=/opt/scripts/crontab_scripts
host=$(hostname -s | tr '[:upper:]' '[:lower:]')
want=()
for f in "$SRC"/cron* "$SRC"/"${host}"_*; do
    [[ -f $f ]] || continue
    name="optscripts-$(basename "$f")"
    install -m 644 "$f" "/etc/cron.d/$name"; want+=("$name")
done
for f in /etc/cron.d/optscripts-*; do
    [[ -e $f ]] || continue
    [[ " ${want[*]} " == *" $(basename "$f") "*" ]] || rm -f "$f"
done
for f in /opt/scripts/profile.d/*.sh; do
    install -C -m 644 "$f" /etc/profile.d/
done
```

Run the installer from its own cron file, so you never overwrite root's personal crontab:

```
# /etc/cron.d/optscripts-installer
*/15 * * * * root /opt/scripts/crontab_scripts/install_cron.sh >/dev/null 2>&1
```

If you put a file in `/var/spool/cron/` instead, it must be mode 0600, or cronie refuses it ("BAD FILE MODE").

Pushing the folder

From the admin box, loop over a host list:

```
while read -r h; do
    rsync -a --delete --chmod=D755,F755 --exclude 'secrets/' /opt/scripts/ "root@$h:/opt/scripts/" \
    || echo "push to $h failed"
done < /opt/scripts/hosts.txt
```

- Keep anything containing secrets **out** of the pushed folder.
- `--chmod` stops loose file modes from being copied everywhere.
- Fix ownership at the source: `chown -R root:root /opt/scripts`.

Why cron0hourly?

The installer above leaves stock files alone. If you use a "replace everything" approach instead, remember that the RHEL `cronie` package's `/etc/cron.d/0hourly` is what runs `/etc/cron.hourly`. Put an equivalent back:

```
01 * * * * root run-parts /etc/cron.hourly
```

Better long-term

Once you have more than a few servers, use Ansible (the `cron` and `copy` modules) or a git pull on each host.

Reliable Let's Encrypt renewals on RHEL / Rocky

A certificate can only be renewed in its last 30 days. A cron job that tries once a month can miss that window after a single failure, and the certificate expires.

Use the packaged timer

The EPEL `certbot` package ships a systemd timer that runs twice a day:

```
sudo systemctl enable --now certbot-renew.timer
systemctl list-timers certbot-renew.timer
```

Remove any old monthly certbot cron entries.

Reload the web server after renewal

A renewed certificate isn't used until the web server reloads. Add a deploy hook:

```
sudo install -m 755 /dev/stdin /etc/letsencrypt/renewal-hooks/deploy/reload-web.sh <<'H'
#!/bin/sh
systemctl reload httpd 2>/dev/null || systemctl reload nginx
H
```

Or, for the packaged timer, set it in `/etc/sysconfig/certbot`:

```
DEPLOY_HOOK="--deploy-hook 'systemctl reload httpd'"
```

If you must use cron

```
17 3,15 * * * root /usr/bin/certbot renew -q --deploy-hook "systemctl reload httpd"
```

Test

```
sudo certbot renew --dry-run
```