



Tâches Planifiées Linux

420-1S6 Systèmes d'exploitation

Tâches planifiées



Parfois, on souhaite que des actions sur notre ordinateur se déroulent **automatiquement** sans que nous ayons besoin d'être présent pour les lancer.

Une **tâche** peut être une ligne de commande ou un script

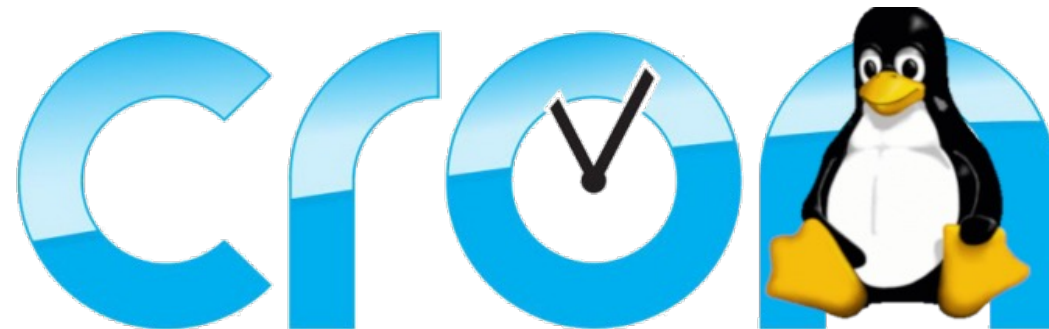
On planifie cette tâche en définissant un **déclencheur** (*trigger*). Par exemple, chaque jour à une heure précise, au démarrage de l'ordinateur, etc.



L'outil cron (Chronos ou Cronos, dieu du temps)



- **cron** est un daemon (service) qui permet d'automatiser l'exécution des tâches.
- Chaque utilisateur possède ses propres tâches planifiées, qui sont configurées dans un fichier nommé **crontab** signifiant Cron Tables.
- Le système possède également son fichier crontab.



Le service cron



Pour gérer le service des tâches planifiées, vous pouvez utiliser :

`systemctl status|start|stop|restart cron`

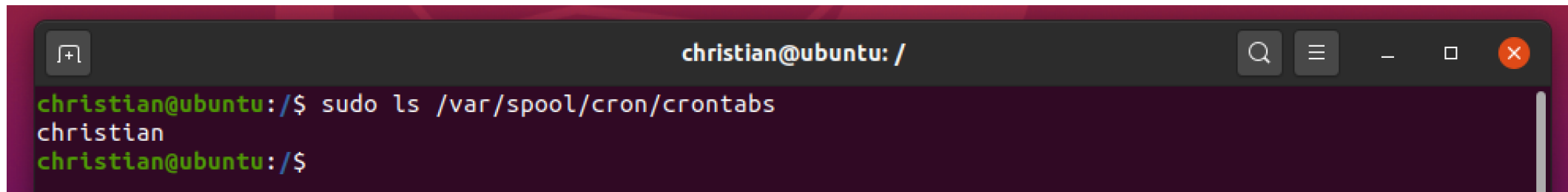
```
christian@ubuntu: ~/Documents/scripts
christian@ubuntu:~/Documents/scripts$ sudo systemctl status cron
● cron.service - Regular background program processing daemon
   Loaded: loaded (/lib/systemd/system/cron.service; enabled; vendor preset: enabled)
   Active: active (running) since Tue 2020-11-17 14:55:01 EST; 24min ago
     Docs: man:cron(8)
  Main PID: 746 (cron)
    Tasks: 1 (limit: 4624)
   Memory: 1.2M
    CGroup: /system.slice/cron.service
            └─746 /usr/sbin/cron -f

Nov 17 14:55:01 ubuntu systemd[1]: Started Regular background program processing daemon.
Nov 17 14:55:01 ubuntu cron[746]: (CRON) INFO (Running @reboot jobs)
Nov 17 15:17:01 ubuntu CRON[3752]: pam_unix(cron:session): session opened for user root by (uid=0)
Nov 17 15:17:01 ubuntu CRON[3761]: (root) CMD ( cd / && run-parts --report /etc/cron.hourly)
Nov 17 15:17:01 ubuntu CRON[3752]: pam_unix(cron:session): session closed for user root
christian@ubuntu:~/Documents/scripts$
```

Fonctionnement de cron



cron recherche dans le répertoire `/var/spool/cron/crontabs` si des fichiers **crontab** existe. Ce répertoire contiendra 1 fichier différent pour chaque utilisateur qui a au moins 1 tâche planifiée de configurée sur le système.



```
christian@ubuntu: /  
christian@ubuntu:/$ sudo ls /var/spool/cron/crontabs  
christian  
christian@ubuntu:/$
```

Ensuite, **cron** se réveille toutes les minutes, examine les fichiers **crontab**, et vérifie chaque commande pour savoir s'il doit la lancer dans la minute à venir.

Ainsi, **cron** n'a pas besoin d'être redémarré après la modification d'un fichier **crontab**.



Visualisation et édition du crontab

Pour voir la liste des tâches planifiées de l'utilisateur courant:

```
crontab -l # (L minuscule)
```

Éditer son propre fichier crontab

```
crontab -e
```

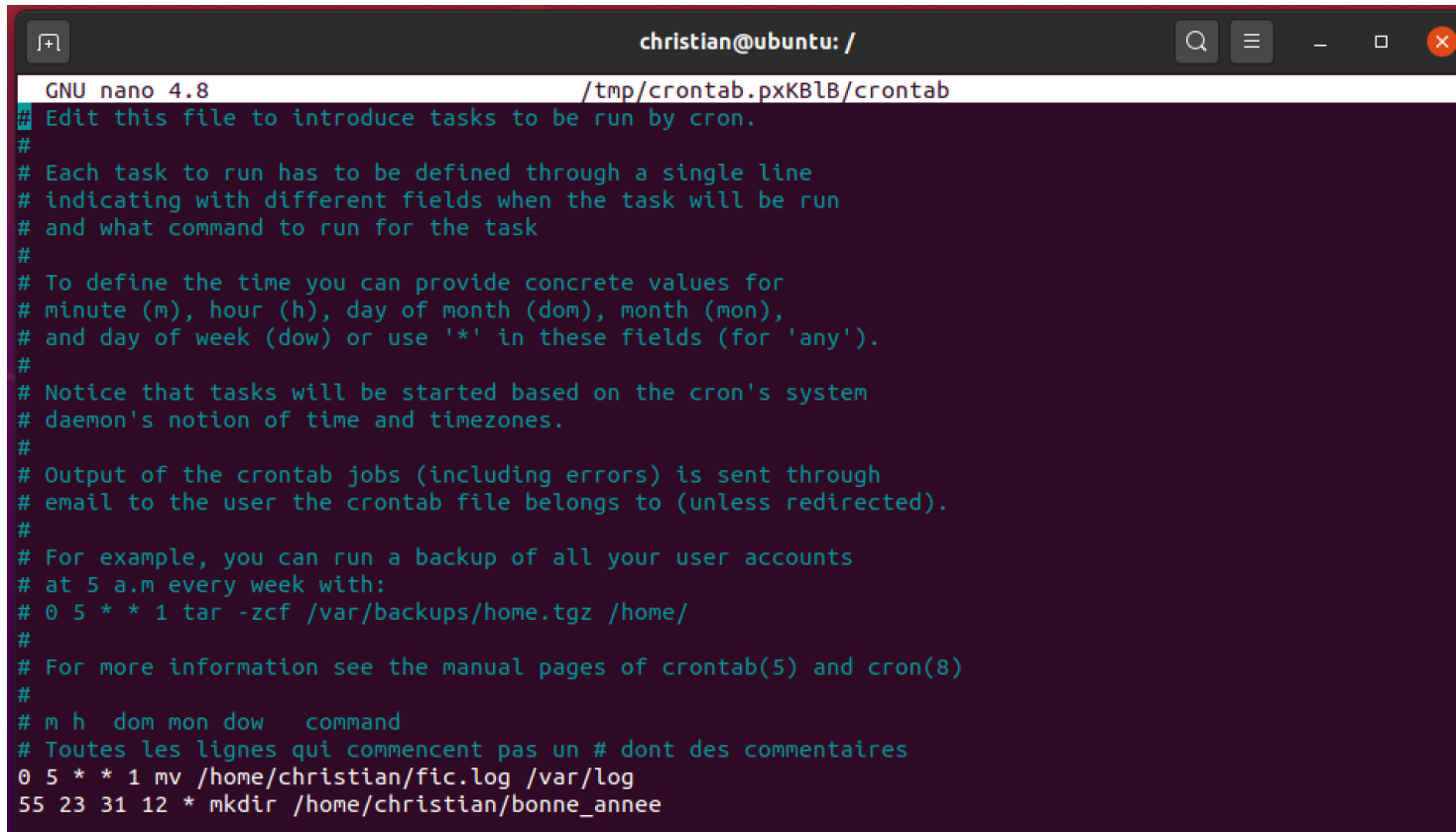
Éditer le fichier crontab de l'utilisateur bob

```
sudo crontab -u bob -e
```

Fichier crontab



Voici l'exemple du contenu d'un fichier crontab
(1 ligne = 1 tâche planifiée)



```
GNU nano 4.8 /tmp/crontab.pxKBlB/crontab
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h dom mon dow   command
# Toutes les lignes qui commencent pas un # dont des commentaires
0 5 * * 1 mv /home/christian/fic.log /var/log
55 23 31 12 * mkdir /home/christian/bonne_annee
```

Fichier crontab – explication de la planification



Commande pour exécuter backup.sh deux fois par jour (à 2h30 et 14h30) du lundi au vendredi
`30 2,14 * * 1-5 root /usr/local/sbin/backup.sh`

| | | | | | | |
|------------|--------------|------|------|------------|------|--|
| 30 0-59 | 2,14 0-23 | * | * | 1-5 0-7 | root | /usr/local/sbin/backup.sh |
| | | 1-31 | 1-12 | | | Commande à exécuter |
| | | | | | | Utilisateur qui exécute la commande |
| | | | | | | Jour de la semaine |
| | | | | | | Dimanche = 0 ou 7; lundi = 1; mardi = 2; mercredi = 3; jeudi = 4; vendredi = 5; samedi = 6 |
| | | | | | | Mois |
| | | | | | | Janvier = 1 (...) Décembre = 12 |
| | | | | | | Jour |
| | | | | | | Heure |
| | | | | | | Minute |

<http://jhroy.ca>

| Colonne | Description |
|------------------|---|
| minute | Minute où sera exécutée la tâche nombre entier entre 0 et 59 |
| heure | Heure où sera exécutée la tâche nombre entier entre 0 et 23 |
| jour | Jour du mois où sera exécutée la tâche Nombre entier entre 1 et 31 |
| mois | Mois où sera exécuté la tâche nombre entier entre 1 et 12 (ou le nom ANSI sur trois lettres ex: 7 ou jul). (1,2,3,4,5,6,7,8,9,10,11,12) <i>[Jan, Feb, Mar, Apr, May, Jun, Jul, Aug, Sep, Oct, Nov, Dec]</i> |
| jourdela semaine | Jour de la semaine où sera exécuté la tâche nombre entier entre 0 et 7 (0 et 7 désignent dimanche, 0=Dimanche, 1= Lundi, 2 = Mardi, etc.) ou le nom ANSI sur trois lettres: <i>Mon, Tue, Wed, Thu, Fri, Sat, Sun</i> |
| *utilisateur | Utilisateur qui exécute la commande (ATTENTION crontab système seulement) |
| commande | Commande ou script à exécuter. |

1) Par Groov3 — Travail personnel, CC BY-SA 4.0,

<https://commons.wikimedia.org/w/index.php?curid=41278749>

2) <https://fr.wikipedia.org/wiki/Cron>



Fichier crontab – valeurs de config

- * : Veut dire répétition pour toutes les valeurs valides
- : Indique une étendue de temps. Ex: dans les minutes: 10-15 veut dire aux minutes 10, 11, 12, 13, 14 et 15.
- , : Indique une liste de valeurs: 15,30 aux minutes signifie à la minute 15 et à la minute 30
- / : Spécifie des valeurs échelonnées. Ex : dans les minutes */5 veut dire une minute sur cinq, ou toutes les 5 minutes.



Fichier crontab – exemples

Tous les jours à 10:00

```
00 10 * * * commande
```

Tous les jours à 20:00

```
00 20 * * * commande
```

Tous les jours à 20:32

```
32 20 * * * commande
```

À toutes les minutes

```
* * * * * commande
```

À tous les jours ouvrables à minuit

```
00 00 * * 1-5 command
```



Fichier crontab – Crontab système

En plus des cron tables pour chaque utilisateur, il existe une notion de crontab système. Celle-ci permet de planifier des tâches d'administration et de spécifier avec quel utilisateur ces tâches seront exécutées.

Le fichier crontab système est /etc/crontab.

run-parts exécute tous les scripts et programmes présents dans le répertoire passé en argument

```
christian@ubuntu: /
christian@ubuntu:/$ cat /etc/crontab
# /etc/crontab: system-wide crontab
# Unlike any other crontab you don't have to run the `crontab`
# command to install the new version when you edit this file
# and files in /etc/cron.d. These files also have username fields,
# that none of the other crontabs do.

SHELL=/bin/sh
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin

# Example of job definition:
# ..... minute (0 - 59)
# | ..... hour (0 - 23)
# | | ..... day of month (1 - 31)
# | | | ..... month (1 - 12) OR jan,feb,mar,apr ...
# | | | | ..... day of week (0 - 6) (Sunday=0 or 7) OR sun,mon,tue,wed,thu,fri,sat
# | | | | |
# * * * * * user-name command to be executed
17 * * * * root cd / && run-parts --report /etc/cron.hourly
25 6 * * * root test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.daily )
47 6 * * 7 root test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.weekly )
52 6 1 * * root test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.monthly )
#
christian@ubuntu:/$
```





Fichier crontab – Crontab système

Attention !

Le fichier `/etc/crontab` contient une colonne de plus : l'utilisateur qui sera utilisé pour exécuter la commande.

Vous devez donc inscrire :

1. Les minutes
2. L'heure
3. Le jour du mois
4. Le mois
5. Le jour de la semaine
6. Le nom d'utilisateur
7. La commande à exécuter

```
christian@ubuntu: /  
christian@ubuntu:/$ cat /etc/crontab  
# /etc/crontab: system-wide crontab  
# Unlike any other crontab you don't have to run the `crontab`  
# command to install the new version when you edit this file  
# and files in /etc/cron.d. These files also have username fields,  
# that none of the other crontabs do.  
  
SHELL=/bin/sh  
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin  
  
# Example of job definition:  
# ----- minute (0 - 59)  
# | ----- hour (0 - 23)  
# | | ----- day of month (1 - 31)  
# | | | ----- month (1 - 12) OR jan,feb,mar,apr ...  
# | | | | ----- day of week (0 - 6) (Sunday=0 or 7) OR sun,mon,tue,wed,thu,fri,sat  
# | | | | |  
# * * * * * user-name command to be executed  
17 * * * * root cd / && run-parts --report /etc/cron.hourly  
25 6 * * * root test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.daily )  
47 6 * * 7 root test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.weekly )  
52 6 1 * * root test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.monthly )  
#  
christian@ubuntu:/$
```





Cron - Journalisation

Les évènements générés par cron sont enregistrés dans le journal syslog.

```
christian@ubuntu: /
christian@ubuntu:/$ sudo cat /var/log/syslog | grep cron
Nov 17 15:00:02 ubuntu anacron[738]: Job `cron.daily' started
Nov 17 15:00:02 ubuntu anacron[3510]: Updated timestamp for job `cron.daily' to 2020-11-17
Nov 17 15:00:02 ubuntu anacron[738]: Job `cron.daily' terminated
Nov 17 15:00:02 ubuntu anacron[738]: Normal exit (1 job run)
Nov 17 15:00:02 ubuntu systemd[1]: anacron.service: Succeeded.
Nov 17 15:17:01 ubuntu CRON[3761]: (root) CMD ( cd / && run-parts --report /etc/cron.hourly)
Nov 17 15:23:52 ubuntu crontab[3796]: (christian) BEGIN EDIT (christian)
Nov 17 15:24:20 ubuntu crontab[3796]: (christian) REPLACE (christian)
Nov 17 15:24:20 ubuntu crontab[3796]: (christian) END EDIT (christian)
Nov 17 15:25:36 ubuntu crontab[3871]: (christian) LIST (christian)
Nov 17 15:30:01 ubuntu CRON[3877]: (root) CMD ([ -x /etc/init.d/anacron ] && if [ ! -d /run/systemd/system ]; then /usr/sbin/invoke-rc.d anacron start >/dev/null; fi)
Nov 17 15:32:36 ubuntu systemd[1]: Started Run anacron jobs.
Nov 17 15:32:36 ubuntu anacron[3878]: Anacron 2.3 started on 2020-11-17
Nov 17 15:32:36 ubuntu anacron[3878]: Normal exit (0 jobs run)
Nov 17 15:32:36 ubuntu systemd[1]: anacron.service: Succeeded.
Nov 17 15:33:20 ubuntu crontab[3893]: (christian) BEGIN EDIT (christian)
Nov 17 16:03:04 ubuntu crontab[3893]: (christian) REPLACE (christian)
Nov 17 16:03:04 ubuntu crontab[3893]: (christian) END EDIT (christian)
Nov 17 16:04:01 ubuntu cron[746]: (christian) RELOAD (crontabs/christian)
christian@ubuntu:/$
```

MTA – Mail Transfert Agent



Lorsqu'une commande exécutée par cron génère un « output » (ex. une confirmation en terminal, une erreur, etc.) cron va tenter d'envoyer la sortie au système de courriel interne de Linux, lorsqu'un tel système est installé.

Sur Ubuntu, il se peut que l'erreur suivante apparaisse : NO MTA INSTALLED, DISCARDING OUTPUT.

```
etudiant@etudiant-virtual-machine:~$ sudo tail -5 /var/log/syslog | grep -i cron
Nov 18 17:36:01 etudiant-virtual-machine CRON[2774]: (CRON) info (No MTA installed, discarding output)
etudiant@etudiant-virtual-machine:~$
```

Pour éviter que la commande exécutée par cron génère une sortie, on peut ajouter `>/dev/null 2>&1` à la fin d'une tâche planifiée afin d'éliminer la sortie.

```
Fichier  Édition  Affichage  Rechercher  Terminal  Aide
GNU nano 2.9.3
* * * * * mv /patates.txt /home/bob >/dev/null 2>&1
```

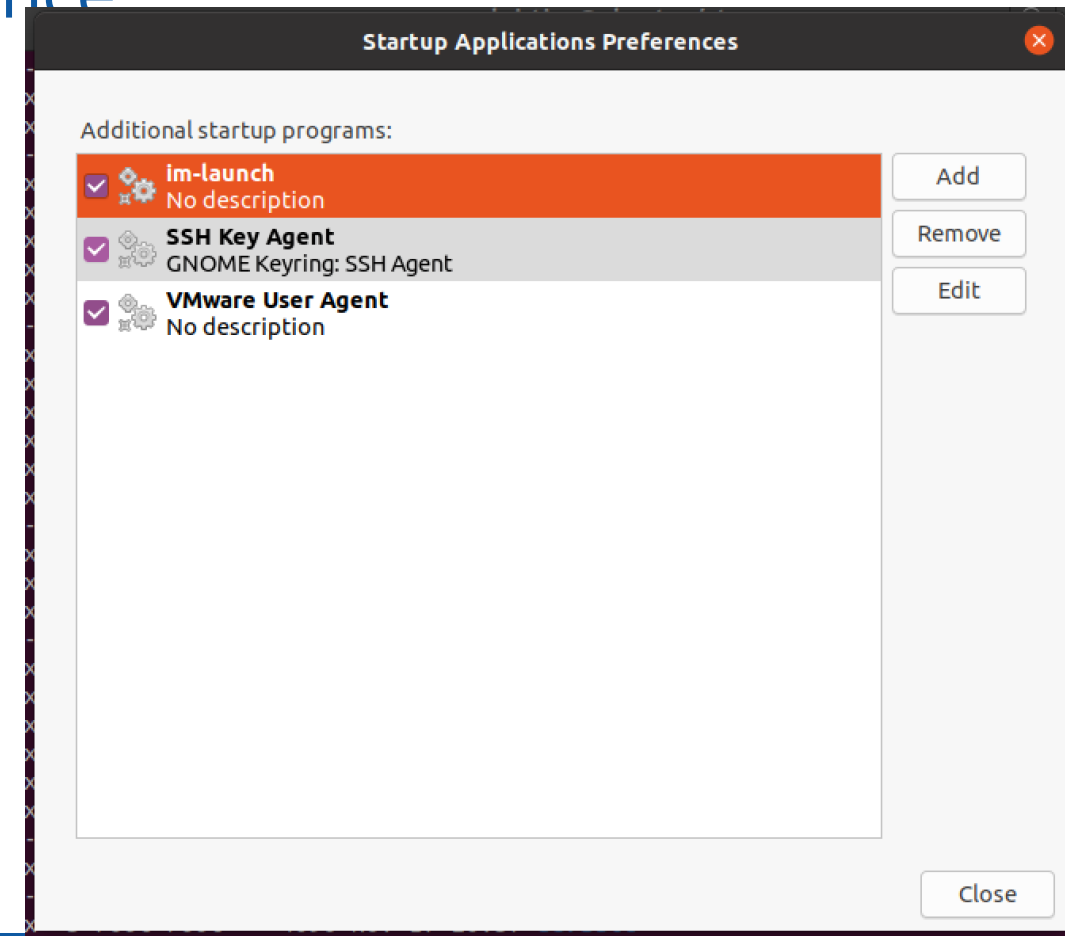
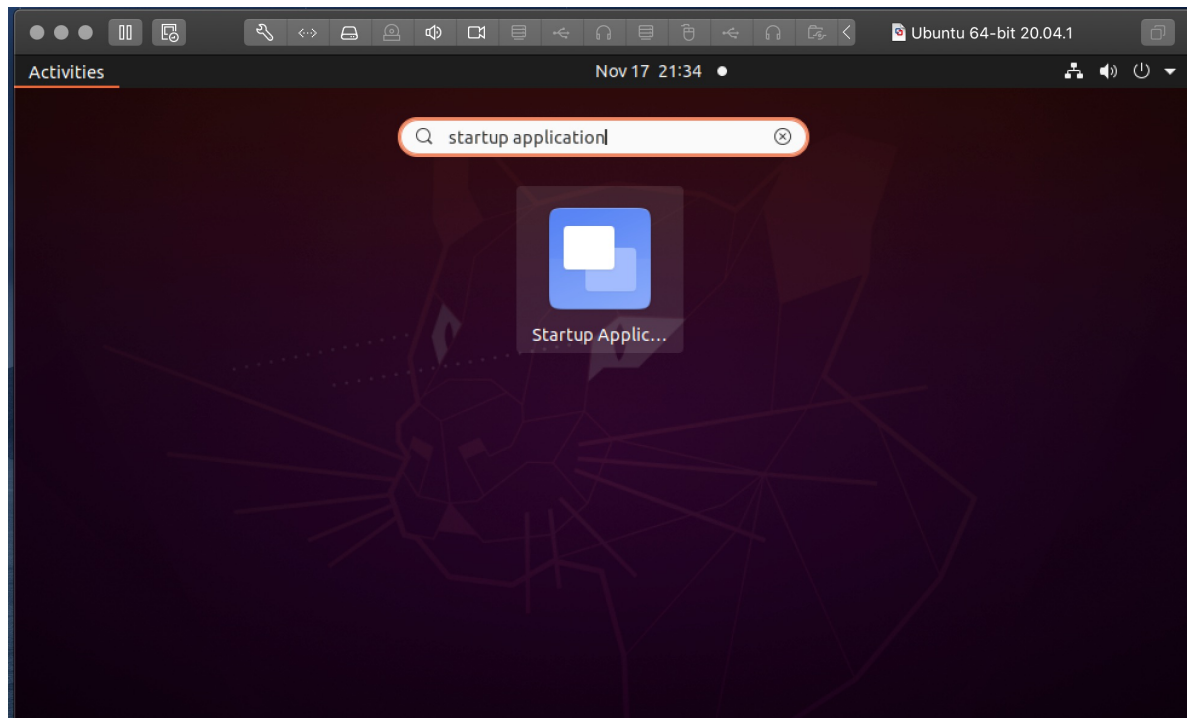


Tâches au démarrage

Séquence de démarrage – Startup application preference



Il est possible d'ajouter des applications à lancer au démarrage via l'application "Startup Application Preference"





Séquence de démarrage – cron

Il est possible de lancer une application au démarrage via cron. Il suffit d'utiliser @reboot, au lieu d'entrer un horaire.

```
rej@rej-vm: ~  
GNU nano 6.2 /etc/crontab *  
# | | | | |  
# * * * * * user-name command to be executed  
17 * * * * root cd / && run-parts --report /etc/cron.hourly  
25 6 * * * root test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.daily )  
47 6 * * 7 root test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.weekly )  
52 6 1 * * root test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.monthly )  
#  
@reboot /home/christian/Documents/scripts/login.sh  
  
^G Aide ^O Écrire ^W Chercher ^K Couper ^T Exécuter ^C Emplacement M-U Annuler M-A Marquer  
^X Quitter ^R Lire fich. ^\ Remplacer ^U Coller ^J Justifier ^/ Aller ligne M-E Refaire M-6 Copier
```