



LES SERVICES LINUX

“DAEMONS”

Réalisé par :
Yamna MOSTAPHI
Kawthar ASSERRAR

Encadré par :
Mr Mehdi MOUKHAFI

SERVICES/DAEMON

Qu'est ce que les services et daemon sur Linux?

01

SYSTEMD

Qu'est ce que systemd dans Linux ?

02

LES UNITS SYSTEMD

Quels sont les units systemd ?

03

TABLE DE MATIERES

04

SYSTEMD/SYSTEMCTL

Comment gérer et configurer les services system avec systemctl ?

05

CREATION D'UN DAEMON

Comment ajouter ou créer un service et daemon dans Linux ?

06

LES JOURNAUX DES SERVICES

Les journaux des services et daemon Linux

01

SERVICES/DAEMONS


Qu'est ce que les services et daemon sur Linux?






SERVICES/DAEMONS

Pour rappel, sous Linux, chaque programme qui s'exécute prend la forme d'un **processus**. On appelle ces processus des **daemons**, qu'on traduit parfois des "demons" en français. Dans le monde Windows, on parle de **Services**. La plupart des logiciels fonctionnent en mode **serveur** ont besoin d'installer un **Daemon**.

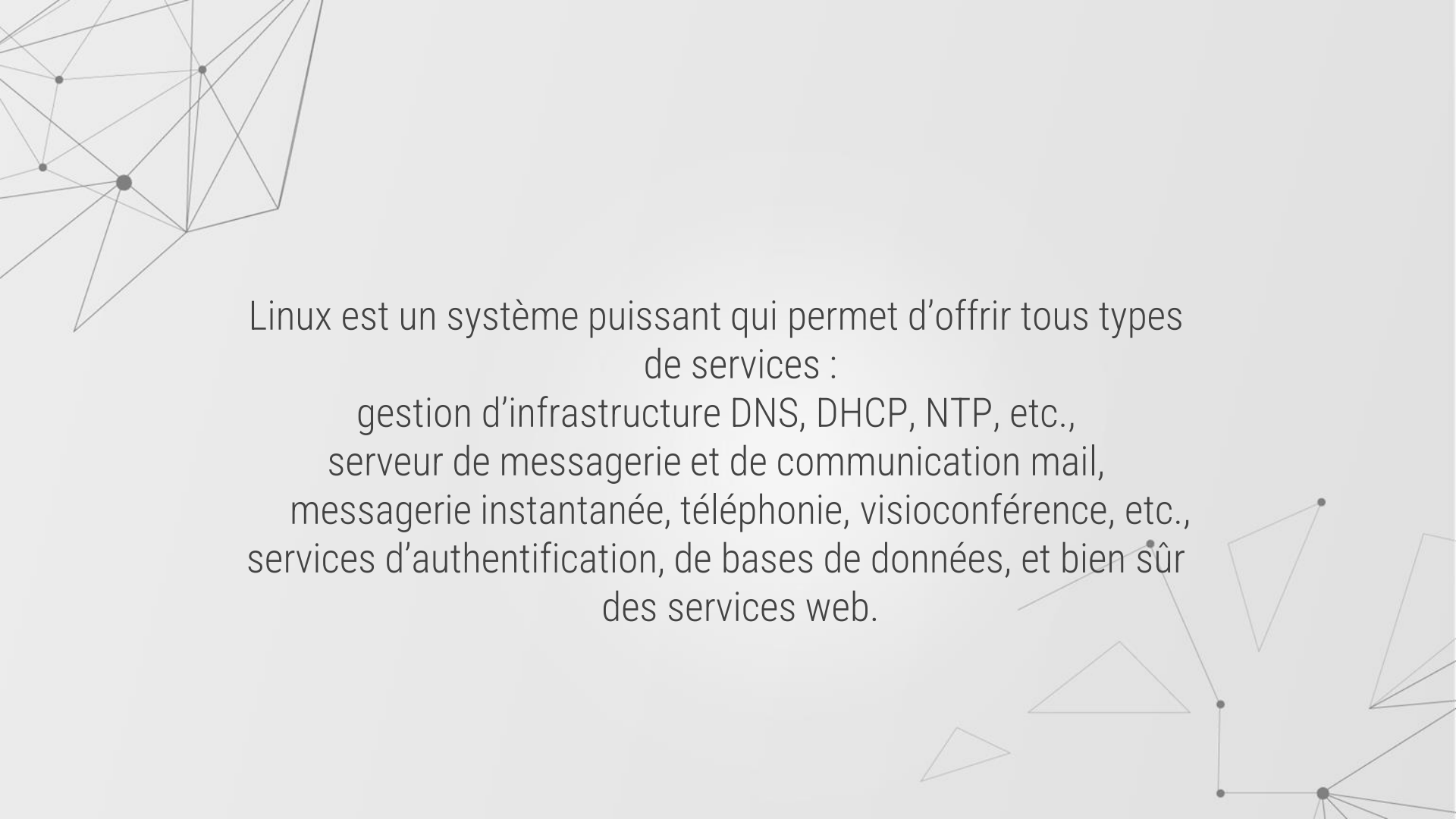




[Un demon] désigne un type de programme informatique, un processus qui s'exécute en arrière-plan plutôt que sous le contrôle direct d'un utilisateur.

Demon signifie : Disk and Execution MONitor (on peut utiliser aussi l'écriture Daemon pour Disk And Execution MONitor).





Linux est un système puissant qui permet d'offrir tous types de services :

- gestion d'infrastructure DNS, DHCP, NTP, etc.,
- serveur de messagerie et de communication mail,
- messagerie instantanée, téléphonie, visioconférence, etc.,
- services d'authentification, de bases de données, et bien sûr des services web.

Types de services

Du côté des services, il existe plusieurs types de service. C'est défini dans le fichier de configuration par **Type=** avec comme valeur possible : simple, forking, oneshot, dbus, notify.

- Service **simple** : C'est le type par défaut. Il lance un processus principal. Le créateur du service doit s'assurer de créer les canaux de communication ou de lancer les processus avant le lancement du dit service. Systemd ne se préoccupe pas de la fin de l'exécution du service pour traiter d'autres unités.
- Service **oneshot** : Le fonctionnement est similaire au type simple. Cependant, systemd attend que le processus soit terminé avant de continuer ses traitement. (Fonctionnement similaire des services sysVinit, au rc.local).
- Service **forking** : Ce service lance un processus père, qui créera un processus fils. Le processus parent s'arrête une fois le service complètement démarré (canaux de communication inclus). Le processus fils tourne tant que le service est démarré. Systemd traite les autres unités une fois le processus père précédemment décrit est terminé. Par analogie, ce sont ainsi que fonctionnent les scripts unix traditionnels.

Il existe d'autres types, tel que notify, et dbus par exemple non abordés ici.

Syntaxe d'un service

Pour les services voici une structure classique, avec quelques options facultatives aussi :

Exemple d'un service

vi /etc/systemd/system/rclocal.service

```
[Unit]
Description=Exécuter le fichier rc.local
ConditionFileIsExecutable=/etc/rc.local
After=network.target
[Service]
Type=oneshot
ExecStart=/etc/rc.local
RemainAfterExit=yes
[Install]
WantedBy=multi-user.target
```

```
[Unit]
Description=
After=
ConditionPathExists=
```

```
[Service]
Type=
ExecStartPre=
ExecStart=
ExecStop=
ExecStopPost=
RemainAfterExit=
Restart=
```

```
[Install]
WantedBy=
```




02

SYSTEMD

Qu'est ce que systemd dans Linux ?



SYSTEMD

Les services sont des scripts d'initialisation gérés initialement par SystemV, depuis Systemd le remplace .

Systemd est un ensemble de logiciels et systèmes nécessaires au fonctionnement Linux.

Notamment ils exposent les daemons : systemd, journald, networkd, logind.

Pour chaque système, un ensemble d'utilitaires et de commandes sont disponibles pour l'utilisateur comme systemctl, journalctl, loginctl, etc.



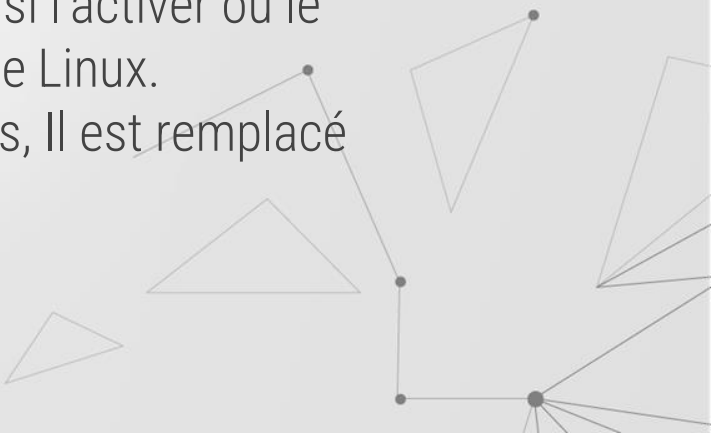


SysV init est l'ancienne suite logicielle de gestion des services et daemon Linux.

Il gère la cohérence entre les services dont les dépendances, les initialise au démarrage de Linux et permet à l'utilisateur de les gérer.

Par exemple, l'utilisateur peut démarrer, arrêter, redémarrer un service mais aussi l'activer ou le désactiver du démarrage Linux.

Dans les distributions Linux récentes, Il est remplacé par systemd.



Ainsi **Systemd** gère l'ensemble des services et daemon Linux.
Il gère de la cohérence et dépendances des services ainsi que leurs initialisation au démarrage Linux.

Systemd utilise un concept d'unit, qui peut être un service, sockets, point de montage, périphériques, etc.

Les fichiers de configuration sont stockés dans **/lib/systemd/system/**.

Les fichiers des services comportent une extension **.service** alors que les sockets ont une extension **.socket**.

```
debian@linux:~$ ls /lib/systemd/system/
apparmor.service          e2scrub_all.timer        kmod.service            paths.target            runlevel5.target.wants  systemd-fsck-root.service
apt-daily.service         e2scrub_fail@.service    kmod-static-nodes.service  php7.4-fpm.service     runlevel6.target        systemd-fsck@.service
apt-daily.timer           e2scrub_reap.service     local-fs-pre.target       phpsessionclean.service  screen-cleanup.service  systemd-halt.service
apt-daily-upgrade.service emergency.service         local-fs.target           phpsessionclean.timer   serial-getty@.service   systemd-hibernate-resume@.service
apt-daily-upgrade.timer  exim4-base.target        logrotate.service         poweroff.target         shutdown.target         systemd-hibernate.service
autovt@.service          exim4-base.timer        logrotate.timer          printer.target          sipgw.target            systemd-hostnamed.service
basic.target             exim4-base.timer        lvm2-lvmpolld.service     procps.service         sleep.target            systemd-hwdb-update.service
blk-availability.service final.target              lvm2-monitor.service     proc-sys-fs-binfmt_misc.automount  slices.target           systemd-hybrid-sleep.service
blockdev@.target         first-boot-complete.target  lvm2-lvmpolld.socket    proc-sys-fs-binfmt_misc.mount  smartcard.target       systemd-initctl.service
bluetooth.target        fstrim.service           lvm2-pvscan@.service     rc-local.service        sound.target            systemd-initctl.socket
boot-complete.target     fstirm.timer             lvm2.service             rc-local.service.d      ssh.service             systemd-journald-audit.socket
cloud-config.service     getty-pre.target         mdadm-grow-continue@.service  rc.service              sshd.service            systemd-journald-dev-log.socket
cloud-init.target        getty@.service           mdadm-last-resort@.service  rc.socket               sudo.service            systemd-journald@.service
cloud-init-local.service getty-static.service      mdadm-last-resort.timer    remote-cryptsetup.target  systemd-journald.socket  systemd-journald@.socket
cloud-init.service       getty.target             mdadm-shutdown.service     reboot.target            sudo.service            systemd-journald@.socket
cloud-init.target        getty.target.wants        mdadm-waitidle.service     remote-fs-pre.target     suspend-target          systemd-journal-flush.service
console-getty.service    graphical.target          mdcheck.continue.service   remote-fs-pre.target     swap.target             systemd-ksm.service
container-getty@.service halt.target              mdcheck_continue.timer     rescue.service           sys-kernel-debug.mount  systemd-located.service
cron.service             hibernate.target         mdcheck_start.service      rescue.target            sys-kernel-fuse-connections.mount  systemd-located.service.d
cryptdisks-early.service hibernat@.service        mdmonitor_oneshot.timer    resolvconf.target.wants  sys-kernel-tracing.mount  systemd-logind.service
cryptdisks-pre.service  hwclock.service          mdmonitor_oneshot.timer    resolvconf.target        sys-kernel-tracing.mount  systemd-machine-id-commit.service
cryptsetup.target       hybrid-sleep.target      mdmonitor_oneshot.timer    resolvconf.service       sys-kernel-tracing.mount  systemd-modules-load.service
ctrl-alt-del.target     ifupdown-pre.service     mdmonitor.service         rsync.service            syslog.socket            systemd-networkd.service
dbus-org.freedesktop.hostname1.service ifupdown-wait-online.service  mdmon@.service            rsync.service            systemd-networkd.socket  systemd-networkd-wait-online.service
dbus-org.freedesktop.local1.service ifupdown-wait-online.service  mdprobe@.service          rsync.service            systemd-network-generator.service
dbus-org.freedesktop.local2.service ifupdown-wait-online.service  multi-user.target.wants   runlevel0.target         systemd-poweroff.service
dbus-org.freedesktop.timedate1.service ifupdown-wait-online.service  multi-user.target.wants   runlevel1.target         systemd-pstore.service
dbus.service             ifupdown-wait-online.service  networking.service        runlevel1.target.wants   systemd-quotacheck.service
dbus.socket              ifupdown-wait-online.service  network-online.target     runlevel2.target         systemd-random-seed.service
debug-shell.service      ifupdown-wait-online.service  network-pre.target        runlevel2.target.wants   systemd-reboot.service
default.target           ifupdown-wait-online.service  network.target            runlevel3.target         systemd-remount-fs.service
dev-hugepages.mount     ifupdown-wait-online.service  nginx.service             runlevel3.target.wants   systemd-resolved.service
dev-mqueue.mount        ifupdown-wait-online.service  nss-lookup.target        runlevel4.target.wants   systemd-rfkill.service
dm-event.service         ifupdown-wait-online.service  nss-user-lookup.target   runlevel5.target         systemd-rfkill.socket
dm-event.socket          ifupdown-wait-online.service  nss-user-lookup.target   runlevel5.target.wants  systemd-suspend.service
e2scrub_all.service      ifupdown-wait-online.service  nss-user-lookup.target   runlevel5.target.wants  systemd-suspend.service
debian@linux:~$
```



Par exemple le service CUPS possède deux fichiers :

- **/lib/systemd/system/cups.service**
- **/lib/systemd/system/cups.socket**

Le fichier du service stocke le nom, description, l'emplacement du fichier de configuration, les commandes à utiliser pour démarrer ou arrêter le service. Du côté du fichier de configuration du socket, on y trouve le nom du service associé et le port en écoute.

```
[kawtharasserrar@fedora ~]$ cat /lib/systemd/system/cups.service
[Unit]
Description=CUPS Scheduler
Documentation=man:cupsd(8)
After=network.target nss-user-lookup.target nslcd.service
Requires=cups.socket

[Service]
ExecStart=/usr/sbin/cupsd -l
Type=notify
Restart=on-failure

[Install]
Also=cups.socket cups.path
WantedBy=printer.target multi-user.target
[kawtharasserrar@fedora ~]$ cat /lib/systemd/system/cups.socket
[Unit]
Description=CUPS Scheduler
PartOf=cups.service

[Socket]
ListenStream=/run/cups/cups.sock

[Install]
WantedBy=sockets.target
[kawtharasserrar@fedora ~]$
```

Les fichiers de configuration **systemd** sont modifiables avec n'importe quel éditeur de texte, comme souvent avec les fichiers de configuration Linux.

Mais on peut aussi utiliser la commande `systemctl` qui ouvrira le fichier de configuration de l'unité dans l'éditeur de texte par défaut.

Pour cela, on utilise la commande `edit` comme ceci :

- **`sudo systemctl edit [nom service]`**

```
GNU nano 6.0 /etc/systemd/system/cups.service.d/.#override.conf2466e7d85d73ad15
## Editing /etc/systemd/system/cups.service.d/override.conf
## Anything between here and the comment below will become the new contents of the file

## Lines below this comment will be discarded

## /usr/lib/systemd/system/cups.service
# [Unit]
# Description=CUPS Scheduler
# Documentation=man:cupsd(8)
# After=network.target nss-user-lookup.target nslcd.service
# Requires=cups.socket
#
# [Service]
# ExecStart=/usr/sbin/cupsd -l
# Type=notify
# Restart=on-failure
#
# [Install]
# Also=cups.socket cups.path
# WantedBy=printer.target multi-user.target
```

[Lecture de 23 lignes]

^G Aide	^O Écrire	^W Chercher	^K Couper	^T Exécuter	^C Emplacement	M-U Annuler	M-A Marquer
^X Quitter	^R Lire fich.	^L Remplacer	^U Coller	^J Justifier	^/ Aller ligne	M-E Refaire	M-G Copier



03

LES UNITS SYSTEMD

Quels sont les units systemd ?

Voici la liste des unités de systemd :

Type d'unité	Description
Automount	Points de montage automatique
Device	Noms de périphérique du noyau, que vous pouvez voir dans SYSFS et UDev
Mount	Les points de montage
Path: file or directory	Fichier et répertoire
Scope	Processus externes non démarrés par systemd
Slice	Une unité de gestion de processus
Snapshot	Sauvegarde des états de snapdpts
Socket	IPC (inter-process communication) socket
Swap	Fichier du swap
Timer	Minuterie système




Pour lister les unités possibles, utilisez la commande `systemctl` de cette manière :

- **`systemctl -t help`**

Enfin utilisez la commande `systemctl` suivante pour lister les unités installés dans votre système Linux :

- **`systemctl list-units`**

```
[kawtharasserrar@fedora ~]$ systemctl -t help
Available unit types:
service
mount
swap
socket
target
device
automount
timer
path
slice
scope
[kawtharasserrar@fedora ~]$
```



04

SYSTEMD/SYSTEMCTL

Comment gérer et configurer les services system avec systemctl ?





SYSTEMD/SYSTEMCTL

systemctl est la commande pour gérer les services Linux.

L'outil **systemctl** permet de configurer les services qui sont lancés au démarrage.



Quand on lance **systemctl** sans aucun paramètre, la liste des daemon et service s'affiche.

Dans la liste se trouve aussi le statut du daemon.

```
systemd-journald.service loaded active running Journal Service
systemd-modules-load.service failed failed Load Kernel Modules
systemd-random-seed.service loaded active exited Load/Save Random Seed
systemd-remount-fs.service loaded active exited Remount Root and Kernel File Systems
systemd-setup-dgram-qlen.service loaded active exited Increase datagram queue length
systemd-sysctl.service loaded active exited Apply Kernel Variables
systemd-tmpfiles-setup-dev.service loaded active exited Create Static Device Nodes in /dev
systemd-tmpfiles-setup.service loaded active exited Create Volatile Files and Directories
systemd-udev-settle.service loaded active exited udev Wait for Complete Device Initialization
systemd-udev-trigger.service loaded active exited udev Coldplug all Devices
systemd-udevd.service loaded active running udev Kernel Device Manager
systemd-update-utmp.service loaded active exited Update UTMPT about System Boot/Shutdown
systemd-user-sessions.service loaded active exited Permit User Sessions
udev-finish.service loaded failed failed Copy rules generated while the root was ro
-.slice loaded active active Root Slice
system-getty.slice loaded active active system-getty.slice
system-openvpn.slice loaded active active system-openvpn.slice
system-systemd-x2dfsc.slice loaded active active system-systemd-x2dfsc.slice
system.slice loaded active active System Slice
user.slice loaded active active User and Session Slice
acpid.socket loaded active running ACPI Daemon Socket
dm-event.socket loaded active listening Device-mapper event daemon FIFOs
lvm2-lvmetad.socket loaded active listening LVM2 metadata daemon socket
syslog.socket loaded active running Syslog Socket
systemd-initctl.socket loaded active listening /dev/initctl Compatibility Named Pipe
systemd-journald-dev-log.socket loaded active running Journal Socket (/dev/log)
systemd-journald.socket loaded active running Journal Socket
systemd-shutdown.socket loaded active listening Delayed Shutdown Socket
systemd-udevd-control.socket loaded active running udev Control Socket
systemd-udevd-kernel.socket loaded active running udev Kernel Socket
dev-sd4.swap loaded active active /dev/sd4 swap
basic.target loaded active active Basic System
cryptsetup.target loaded active active Encrypted Volumes
getty.target loaded active active Login Prompts
graphical.target loaded active active Graphical Interface
local-fs-pre.target loaded active active Local File Systems (Pre)
local-fs.target loaded active active Local File Systems
mail-transport-agent.target loaded active active Mail Transport Agent
multi-user.target loaded active active Multi-User System
network-online.target loaded active active Network is Online
network.target loaded active active Network
nss-lookup.target loaded active active Host and Network Name Lookups
paths.target loaded active active Paths
remote-fs-pre.target loaded active active Remote File Systems (Pre)
remote-fs.target loaded active active Remote File Systems
slices.target loaded active active Slices
sockets.target loaded active active Sockets
swap.target loaded active active Swap
sysinit.target loaded active active System Initialization
timers.target loaded active active Timers
systemd-tmpfiles-clean.timer loaded active waiting Daily Cleanup of Temporary Directories

LOAD = Reflects whether the unit definition was properly loaded.
ACTIVE = The high-level unit activation state, i.e. generalization of SUB.
SUB = The low-level unit activation state, values depend on unit type.

371 loaded units listed. Pass --all to see loaded but inactive units, too.
To show all installed unit files use 'systemctl list-unit-files'.
lines 322-379/379 (END)
```

Si vous ne désirez lister que les daemons et services actifs alors utilisez la commande systemctl de cette manière :

systemctl list-units --type=service



Configurer ses services

- **Activer un service au démarrage**

Pour configurer un service pour qu'il soit lancé automatiquement au démarrage du système, utiliser la commande :

```
systemctl enable nom_du_service.service
```

- **Désactiver un service au démarrage**

Pour configurer un service pour qu'il ne soit plus lancé automatiquement au démarrage du système, utiliser la commande :

```
systemctl disable nom_du_service.service
```

Services en fonction

- **Statut d'un service**

Pour connaître le statut d'un service, utiliser la commande :

```
# systemctl is-active nom_du_service.service
```

Les informations sont très minimalistes.
Pour plus de détails, utiliser la commande :

```
systemctl status nom_du_service.service
```





Exécution de services

- **Démarrer un service**

Pour démarrer un service, utilisez la commande :

```
# systemctl start nom_du_service.service
```

- **Arrêter un service**

Pour arrêter un service, utilisez la commande :

```
# systemctl stop nom_du_service.service
```

- **Redémarrer un service**

Une commande existe pour redémarrer un service (ce qui correspond à un systemctl stop enchaîné d'un systemctl start) :

```
systemctl restart nom_du_service.service
```

- **Recharger la configuration un service**

On peut recharger la configuration sans interrompre le service avec cette commande :

```
systemctl reload nom_du_service.service
```



05

CREATION D'UN DAEMON

Comment ajouter ou créer un service et daemon dans Linux ?





Comment fonctionnent les services init.d Linux

Le fonctionnement des daemon s'appuie sur plusieurs scripts se trouvant dans le dossier /etc. Cette partie est identique pour la plupart des distributions :

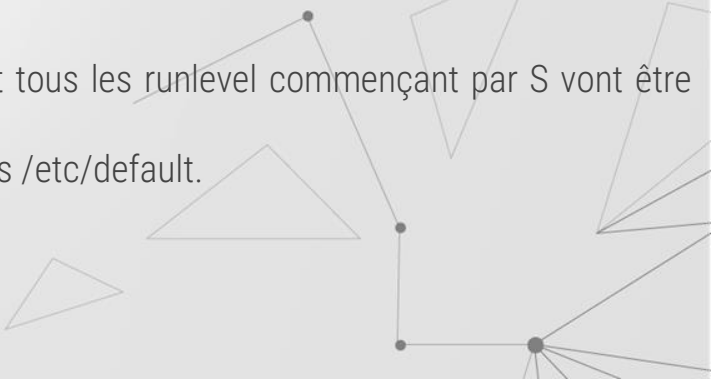
- **/etc/init.d** : ce dernier stocke les scripts de lancement des daemons
- **/etc/default** : stocke les fichiers de configuration des daemons et services. La plupart du temps il s'agit des paramètres de lancement. La configuration a probablement parlé de l'application se trouve dans /etc/
- **/etc/rcX.d** : stocke les scripts runlevel. Ces derniers se déclenchent lors de différents évènements : allumage, arrêt, etc.
- **/etc/inittab** est le fichier de configuration des runlevel

Prenons le cas du démarrage de l'ordinateur.

Le runlevel pour cet évènement se déclenche (souvent /etc/rc2.d) est parsé et tous les runlevel commençant par S vont être démarrés.

Chaque daemon va alors exécuter le script /etc/init.d et lire la configuration dans /etc/default.

Il se passe la même chose lors de l'arrêt de l'ordinateur.



Le répertoire /etc/init.d

Il s'agit des scripts de lancement des daemon et services.

Certains sont liés à des applications installées, d'autres à des utilitaires systèmes.

En général ces derniers portent le nom du service et application.

Par exemple apache2 correspond à l'application Web Apache.

```
.root@ovh-4:~# ls -lh /etc/init.d/
total 352K
-rwxr-xr-x 1 root root 2,2K avril  1 2012 acpid
-rwxr-xr-x 1 root root 7,7K août  18 2015 apache2
-rwxr-xr-x 1 root root 3,4K juil. 30 2012 bind9
-rwxr-xr-x 1 root root 1,3K août 11 2012 bootlogs
-rwxr-xr-x 1 root root 1,3K août 11 2012 bootmisc.sh
-rwxr-xr-x 1 root root 3,8K août 11 2012 checkfs.sh
-rwxr-xr-x 1 root root 1,3K déc. 11 2012 checkroot-bootclean.sh
-rwxr-xr-x 1 root root 9,5K août 11 2012 checkroot.sh
-rwxr-xr-x 1 root root 3,0K juil.  3 2012 cron
-rwxr-xr-x 1 root root 2,8K févr.  5 2015 dbus
-rwxr-xr-x 1 root root 19K févr.  4 10:49 firewall.sh
-rwxr-xr-x 1 root root 1,3K août 11 2012 halt
-rwxr-xr-x 1 root root 3,0K oct.  30 2011 hddtemp
-rwxr-xr-x 1 root root 1,4K août 11 2012 hostname.sh
-rwxr-xr-x 1 root root 3,8K juin  23 2012 hwclock.sh
-rwxr-xr-x 1 root root 2,6K déc.  28 2011 ipmievd
-rwxr-xr-x 1 root root 2,0K juin  16 2012 irqbalance
-rwxr-xr-x 1 root root 1,3K août 11 2012 killprocs
-rwxr-xr-x 1 root root 2,0K mai  20 2012 kmod
-rwxr-xr-x 1 root root 884 mai  4 2012 lvm2
-rwxr-xr-x 1 root root 2,4K janv. 24 2013 mdadm
-rwxr-xr-x 1 root root 6,2K oct.  20 2012 mdadm-raid
-rwxr-xr-x 1 root root 1,2K janv. 24 2013 mdadm-waitidle
-rwxr-xr-x 1 root root 995 août 11 2012 motd
-rwxr-xr-x 1 root root 670 déc.  11 2012 mountall-bootclean.sh
-rwxr-xr-x 1 root root 2,1K janv. 24 2013 mountall.sh
-rwxr-xr-x 1 root root 1,5K août 11 2012 mountdevsubfs.sh
-rwxr-xr-x 1 root root 1,4K août 11 2012 mountkernfs.sh
-rwxr-xr-x 1 root root 678 déc.  11 2012 mountnfs-bootclean.sh
-rwxr-xr-x 1 root root 2,4K août 11 2012 mountnfs.sh
-rwxr-xr-x 1 root root 1,7K août 11 2012 mtab.sh
-rwxr-xr-x 1 root root 5,4K oct.  23 2015 mysql
-rwxr-xr-x 1 root root 4,3K mars  14 2013 networking
-rwxr-xr-x 1 root root 6,4K mai  11 2013 nfs-common
-rwxr-xr-x 1 root root 5,6K août 13 2014 nfs-common.dpkg-new
-rwxr-xr-x 1 root root 1,8K févr.  4 2015 ntp
-rwxr-xr-x 1 root root 8,8K févr. 24 2012 openvpn
-rwxr-xr-x 1 root root 7,2K mars  6 2013 postfix
-rwxr-xr-x 1 root root 1,4K mai  20 2012 procps
-rwxr-xr-x 1 root root 6,0K oct.  15 2012 rc
-rwxr-xr-x 1 root root 782 août 11 2012 rc.local
-rwxr-xr-x 1 root root 117 oct.  15 2012 rcS
-rw-r--r-- 1 root root 2,4K oct.  15 2012 README
-rwxr-xr-x 1 root root 639 août 11 2012 reboot
-rwxr-xr-x 1 root root 1,1K août 11 2012 rnmlogin
-rwxr-xr-x 1 root root 2,3K mai  10 2017 rpbind
-rwxr-xr-x 1 root root 4,3K oct.  4 2011 rsync
-rwxr-xr-x 1 root root 3,0K mai  4 2012 rsyslog
-rwxr-xr-x 1 root root 1,2K juin 21 2012 screen-cleanup
-rwxr-xr-x 1 root root 3,2K août 11 2012 sendsigs
-rwxr-xr-x 1 root root 590 août 11 2012 single
-rw-r--r-- 1 root root 4,2K oct.  15 2012 skeleton
-rwxr-xr-x 1 root root 19K juin 19 2011 smartd
-rwxr-xr-x 1 root root 3,5K juin 19 2011 smartmontools
-rwxr-xr-x 1 root root 3,2K sept. 21 2014 snmpd
-rwxr-xr-x 1 root root 3,8K févr. 24 2012 ssh
```

Ci-dessous, un exemple de script init.d qui permet de lancer apache

```
root@ovh-4:~# cat /etc/init.d/apache2
#!/bin/sh
### BEGIN INIT INFO
# Provides:          apache2
# Required-Start:    $local_fs $remote_fs $network $syslog $named
# Required-Stop:     $local_fs $remote_fs $network $syslog $named
# Default-Start:     2 3 4 5
# Default-Stop:      0 1 6
# X-Interactive:     true
# Short-Description: Start/stop apache2 web server
### END INIT INFO

set -e

SCRIPTNAME="${0##*/}"
SCRIPTNAME="${SCRIPTNAME##[KS][0-9][0-9]}"
if [ -n "$APACHE_CONFDIR" ] ; then
    if [ "${APACHE_CONFDIR##/etc/apache2-}" != "${APACHE_CONFDIR}" ] ; then
        DIR_SUFFIX="${APACHE_CONFDIR##/etc/apache2-}"
    else
        DIR_SUFFIX=
    fi
elif [ "${SCRIPTNAME##apache2-}" != "${SCRIPTNAME}" ] ; then
    DIR_SUFFIX="-${SCRIPTNAME##apache2-}"
    APACHE_CONFDIR=/etc/apache2$DIR_SUFFIX
else
    DIR_SUFFIX=
    APACHE_CONFDIR=/etc/apache2
fi
if [ -z "$APACHE_ENVVARS" ] ; then
    APACHE_ENVVARS=$APACHE_CONFDIR/envvars
fi
export APACHE_CONFDIR APACHE_ENVVARS

ENV="env -i LANG=C PATH=/usr/local/bin:/usr/bin:/bin"
if [ "$APACHE_CONFDIR" != /etc/apache2 ] ; then
    ENV="$ENV APACHE_CONFDIR=$APACHE_CONFDIR"
fi
if [ "$APACHE_ENVVARS" != "$APACHE_CONFDIR/envvars" ] ; then
    ENV="$ENV APACHE_ENVVARS=$APACHE_ENVVARS"
fi

#edit /etc/default/apache2 to change this.
HTCACHECLEAN_RUN=auto
HTCACHECLEAN_MODE=daemon
HTCACHECLEAN_SIZE=300M
HTCACHECLEAN_DAEMON_INTERVAL=120
HTCACHECLEAN_PATH=/var/cache/apache2$DIR_SUFFIX/mod_disk_cache
HTCACHECLEAN_OPTIONS=""

APACHE_HTTPD=$(. $APACHE_ENVVARS && echo $APACHE_HTTPD)
if [ -z "$APACHE_HTTPD" ] ; then
    APACHE_HTTPD=/usr/sbin/apache2
fi
```



Les runlevel (/etc/rcX.d)

Enfin on trouve les runlevel qui s'exécutent à différents états de l'ordinateur selon le numéro de ce dernier.

Tous les runlevel pointent vers le script d'init.d

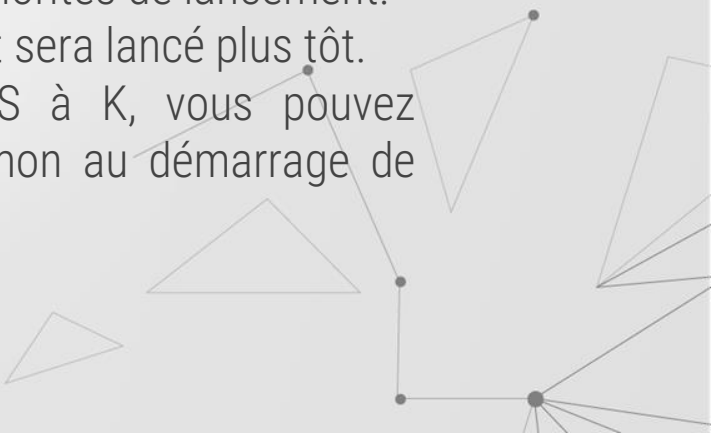
S pour start indique un lancement

K pour Kill tue le daemon

Les numéros derrière S ou K sont les priorités de lancement.

Plus le numéro est petit et plus le script sera lancé plus tôt.

Ainsi en passant un runlevel d'un S à K, vous pouvez désactiver le lancement d'un daemon au démarrage de l'ordinateur.





Par exemple ci-dessous, on voit que halt qui correspond à l'arrêt de l'ordinateur possède le chiffre 0 alors que hddtemp possède 01. Hddtemp sera donc exécuté dans les premiers.

Généralement donc les applications installées ont des chiffres bas pour les runlevel d'arrêt de l'ordinateur et les numéros plus élevés correspondent à des daemon systèmes.


C'est l'inverse pour les runlevel de lancement.

```
doot@ovh-4:~# ls -lh /etc/rc*
-rwxr-xr-x 1 root root 1,6K oct. 28 2015 /etc/rc.local

/etc/rc0.d:
total 4,0K
lrwxrwxrwx 1 root root 17 févr. 10 2014 K01hddtemp -> ../init.d/hddtemp
lrwxrwxrwx 1 root root 17 oct. 29 2015 K01ipmievd -> ../init.d/ipmievd
lrwxrwxrwx 1 root root 15 févr. 10 2014 K01mdadm -> ../init.d/mdadm
lrwxrwxrwx 1 root root 17 oct. 28 2015 K01openvpn -> ../init.d/openvpn
lrwxrwxrwx 1 root root 17 oct. 31 2015 K01postfix -> ../init.d/postfix
lrwxrwxrwx 1 root root 15 juin 9 2018 K01snmpd -> ../init.d/snmpd
lrwxrwxrwx 1 root root 17 févr. 10 2014 K01urandom -> ../init.d/urandom
lrwxrwxrwx 1 root root 17 juin 9 2018 K02apache2 -> ../init.d/apache2
lrwxrwxrwx 1 root root 15 oct. 31 2015 K02mysql -> ../init.d/mysql
lrwxrwxrwx 1 root root 15 juin 9 2018 K03bind9 -> ../init.d/bind9
lrwxrwxrwx 1 root root 18 juin 9 2018 K04sendsigs -> ../init.d/sendsigs
lrwxrwxrwx 1 root root 17 juin 9 2018 K05rsyslog -> ../init.d/rsyslog
lrwxrwxrwx 1 root root 22 juin 9 2018 K06umountnfs.sh -> ../init.d/umountnfs.sh
lrwxrwxrwx 1 root root 20 juin 9 2018 K07nfs-common -> ../init.d/nfs-common
lrwxrwxrwx 1 root root 17 juin 9 2018 K07rpcbind -> ../init.d/rpcbind
lrwxrwxrwx 1 root root 20 juin 9 2018 K08hwclock.sh -> ../init.d/hwclock.sh
lrwxrwxrwx 1 root root 20 juin 9 2018 K08networking -> ../init.d/networking
lrwxrwxrwx 1 root root 18 juin 9 2018 K09umountfs -> ../init.d/umountfs
lrwxrwxrwx 1 root root 20 juin 9 2018 K10mdadm-raid -> ../init.d/mdadm-raid
lrwxrwxrwx 1 root root 20 juin 9 2018 K10umountroot -> ../init.d/umountroot
lrwxrwxrwx 1 root root 24 juin 9 2018 K11mdadm-waitidle -> ../init.d/mdadm-waitidle
lrwxrwxrwx 1 root root 14 juin 9 2018 K12halt -> ../init.d/halt
-rw-r--r-- 1 root root 353 oct. 15 2012 README

/etc/rc1.d:
total 4,0K
lrwxrwxrwx 1 root root 17 févr. 10 2014 K01hddtemp -> ../init.d/hddtemp
lrwxrwxrwx 1 root root 17 oct. 29 2015 K01ipmievd -> ../init.d/ipmievd
lrwxrwxrwx 1 root root 15 févr. 10 2014 K01mdadm -> ../init.d/mdadm
lrwxrwxrwx 1 root root 17 oct. 28 2015 K01openvpn -> ../init.d/openvpn
lrwxrwxrwx 1 root root 17 oct. 31 2015 K01postfix -> ../init.d/postfix
lrwxrwxrwx 1 root root 23 févr. 10 2014 K01smartmontools -> ../init.d/smartmontools
lrwxrwxrwx 1 root root 15 juin 9 2018 K01snmpd -> ../init.d/snmpd
lrwxrwxrwx 1 root root 16 oct. 29 2015 K01vsftpd -> ../init.d/vsftpd
lrwxrwxrwx 1 root root 17 juin 9 2018 K02apache2 -> ../init.d/apache2
lrwxrwxrwx 1 root root 15 oct. 31 2015 K02mysql -> ../init.d/mysql
lrwxrwxrwx 1 root root 15 juin 9 2018 K03bind9 -> ../init.d/bind9
lrwxrwxrwx 1 root root 17 juin 9 2018 K05rsyslog -> ../init.d/rsyslog
lrwxrwxrwx 1 root root 20 juin 9 2018 K07nfs-common -> ../init.d/nfs-common
lrwxrwxrwx 1 root root 17 juin 9 2018 K07rpcbind -> ../init.d/rpcbind
-rw-r--r-- 1 root root 369 oct. 15 2012 README
lrwxrwxrwx 1 root root 19 févr. 10 2014 S01killprocs -> ../init.d/killprocs
lrwxrwxrwx 1 root root 14 févr. 10 2014 S01motd -> ../init.d/motd
lrwxrwxrwx 1 root root 18 déc. 4 2017 S19bootlogs -> ../init.d/bootlogs
lrwxrwxrwx 1 root root 16 déc. 4 2017 S20single -> ../init.d/single

/etc/rc2.d:
total 4,0K
-rw-r--r-- 1 root root 677 juil. 14 2013 README
lrwxrwxrwx 1 root root 14 févr. 10 2014 S01motd -> ../init.d/motd
```

Les actions possibles sont toujours **start** et **stop** pour démarrer et arrêter un processus. Vous pouvez parfois exécuter **restart** (stop puis start en une seule commande), **reload** pour mettre à jour la configuration sans arrêter le service, ou **status** pour savoir si le daemon est actif et si tout fonctionne correctement.


Pour savoir quels services doivent être démarrés au lancement du système, ou à l'inverse pour tout arrêter proprement, SystemV se base sur des runlevels. Il y en a 7, numérotés de 0 à 6, et dont la fonction peut varier légèrement d'une distribution Linux à l'autre. Il y a généralement un consensus sur les niveaux :

Runlevel 0 : pour arrêter le système.

Runlevel 1 : pour le mode "single user", qui est un mode de maintenance dans lequel seules les fonctions essentielles sont démarrées. Il n'y a pas de réseau, et seul root peut se connecter en ligne de commande.

Runlevel 6 : pour redémarrer le système..

Pour **les runlevels 2 à 5**, ce sont des modes multi-utilisateurs graphiques ou en mode console. C'est généralement le mode de fonctionnement normal du système. Le mode par défaut est défini dans le fichier **/etc/inittab** .



Le fichier **/etc/inittab** donne la configuration des runlevel dont notamment celui de démarrage et arrêt de l'ordinateur.

Certaines configurations sont liées à la combinaisons de touches CTRL+ALT+Suppr ou un arrêt de l'ordinateur qui ne se fait pas correctement.

Par défaut sur une distribution Debian, on a les runlevel suivants :

0 : runlevel à l'arrêt de l'ordinateur

1 : singlemode lorsque l'ordinateur se lance en maintenance par exemple pour lancer un fsck.

2-5 : runlevel au lancement de l'ordinateur en mode normal

6: redémarrage via la commande reboot

```
root@ovh-4:~# cat /etc/inittab
# /etc/inittab: init(8) configuration.
# $Id: inittab,v 1.91 2002/01/25 13:35:21 miquels Exp $

# The default runlevel.
id:2:initdefault:

# Boot-time system configuration/initialization script.
# This is run first except when booting in emergency (-b) mode.
si::sysinit:/etc/init.d/rcS

# What to do in single-user mode.
~:S:wait:/sbin/sulogin

# /etc/init.d executes the S and K scripts upon change
# of runlevel.
#
# Runlevel 0 is halt.
# Runlevel 1 is single-user.
# Runlevels 2-5 are multi-user.
# Runlevel 6 is reboot.

l0:0:wait:/etc/init.d/rc 0
l1:1:wait:/etc/init.d/rc 1
l2:2:wait:/etc/init.d/rc 2
l3:3:wait:/etc/init.d/rc 3
l4:4:wait:/etc/init.d/rc 4
l5:5:wait:/etc/init.d/rc 5
l6:6:wait:/etc/init.d/rc 6
# Normally not reached, but fallthrough in case of emergency.
z6:6:respawn:/sbin/sulogin

# What to do when CTRL-ALT-DEL is pressed.
ca:12345:ctrlaltdel:/sbin/shutdown -t1 -a -r now

# Action on special keypress (ALT-UpArrow).
#kb::kbrequest:/bin/echo "Keyboard Request--edit /etc/inittab to let this work."

# What to do when the power fails/returns.
pf::powerwait:/etc/init.d/powerfail start
pn::powerfailnow:/etc/init.d/powerfail now
po::powerokwait:/etc/init.d/powerfail stop

# /sbin/getty invocations for the runlevels.
#
# The "id" field MUST be the same as the last
# characters of the device (after "tty").
#
# Format:
# <id>:<runlevels>:<action>:<process>
#
# Note that on most Debian systems tty7 is used by the X Window System,
# so if you want to add more getty's go ahead but skip tty7 if you run X.
#
1:2345:respawn:/sbin/getty 38400 tty1
2:23:respawn:/sbin/getty 38400 tty2
```

/etc/default

Ce dossier stocke les configurations des services.

Il s'agit souvent de paramètres de lancement pour activer telle ou telle options de l'application. Lorsque vous venez d'installer un daemon, il est donc conseillé de vérifier le contenu de ce fichier.

```
root@ovh-3:~# ls -lh /etc/default/
total 140K
-rw-r--r-- 1 root root 556 nov. 3 12:34 apache-htcacheclean
-rw-r--r-- 1 root root 85 nov. 20 15:37 bind9
-rw-r--r-- 1 root root 85 févr. 17 2016 bind9.dpkg-dist
-rw-r--r-- 1 root root 222 mai 16 2012 bsdmaintools
-rw-r--r-- 1 root root 955 juil. 3 2012 cron
-rw-r--r-- 1 root root 297 oct. 4 2012 dbus
-rw-r--r-- 1 root root 92 nov. 15 2010 devpts
-rw-r--r-- 1 root root 148 févr. 12 2011 fixudev
-rw-r--r-- 1 root root 1,1K févr. 7 2011 grub
-rw-r--r-- 1 root root 1,2K févr. 19 2016 grub.ucf-dist
-rw-r--r-- 1 root root 86 nov. 15 2010 halt
-rw-r--r-- 1 root root 1,2K nov. 20 15:37 hddtemp
-rw-r--r-- 1 root root 657 déc. 11 2012 hwclock
-rw-r--r-- 1 root root 931 nov. 20 15:44 irqbalance
-rw-r--r-- 1 root root 561 nov. 4 2015 keyboard
-rw-r--r-- 1 root root 35 nov. 4 2015 locale
-rw-r--r-- 1 root root 718 janv. 17 19:20 mdadm
-rw-r--r-- 1 root root 1,2K août 10 2017 mysql
-rw-r--r-- 1 root root 306 mars 14 2013 networking
-rw-r--r-- 1 root root 793 mai 11 2013 nfs-common
-rw-r--r-- 1 root root 1,8K févr. 12 2016 nss
-rw-r--r-- 1 root root 15 juil. 13 2010 ntp
-rw-r--r-- 1 root root 540 avril 3 2012 ntpdate
-rw-r--r-- 1 root root 535 nov. 4 2015 openvpn
-rw-r--r-- 1 root root 1,2K nov. 7 2014 openvpn.dpkg-dist
-rw-r--r-- 1 root root 383 févr. 19 2016 rcS
-rw-r--r-- 1 root root 821 févr. 12 2017 rcS.dpkg-dist
-rw-r--r-- 1 root root 1,8K nov. 4 2015 rsync
-rw-r--r-- 1 root root 2,1K déc. 10 2017 rsync.dpkg-dist
-rw-r--r-- 1 root root 124 déc. 14 2015 rsyslog
-rw-r--r-- 1 root root 427 déc. 17 2015 smartmontools
-rw-r--r-- 1 root root 585 oct. 9 10:45 snmpd
-rw-r--r-- 1 root root 133 mars 31 2010 ssh
-rw-r--r-- 1 root root 1,3K févr. 12 2017 tmpfs
-rw-r--r-- 1 root root 1,1K sept. 26 2010 useradd
root@ovh-3:~# cat /etc/default/bind9
# run resolvconf?
RESOLVCONF=yes

# startup options for the server
OPTIONS="-u bind"
root@ovh-3:~#
```



Activer/Désactiver un service

Dans les distributions de type Debian, la commande **update-rc.d** permet de créer ou modifier facilement un runlevel. **update-rc.d** permet donc de configurer les runlevel d'un daemon. En clair cela va installer les liens vers le script System-V. Pour créer les runlevel par défaut, on peut utiliser la commande de cette manière :

```
update-rc.d nom_daemon defaults
```

Pour retirer un script à l'arrêt :


```
update-rc.d -f nom_daemon remove
```

Rdémarrer un daemon ou service

Plusieurs méthodes sont possibles pour relancer un daemon ou service.

Vous pouvez passer par le script, par exemple :


```
/etc/init.d/daemon start  
/etc/init.d/daemon restart  
/etc/init.d/daemon stop
```






Pourquoi faire un demon ?

Évidemment, on pourrait se demander pourquoi faire un demon. Imaginez simplement que vous avez un tout petit serveur pour votre famille, pour vos amis, ou même un serveur de communication vocale. Ce serveur tourne sous Linux et à chaque fois que vous démarrez le serveur, il faut entrer login, mot de passe, puis lancer le terminal, s'identifier en tant que root et – enfin – lancer le script de démarrage. Eh bien le demon permet de lancer un script avec toutes les autorisations nécessaires, avant d'ouvrir sa session.





Pour réaliser un demon, tout se passe dans ces dossiers : /etc/init.d/ et /usr/bin/
Lancez un terminal.

- Entrez-y la commande pour avoir accès aux droits root :
- sudo -s**
- Entrez votre mot de passe (le mot de passe root).
 - Localisez votre script de démarrage et copiez-le dans /usr/bin :

cp /dir1/dir2/launcher /usr/bin/launcher

il faut créer un programme de lancement automatique situé dans /etc/init.d/
C'est là que Linux nous fournit un squelette, un cadre : skeleton.

- Donc copiez le skeleton dans votre script avec :

cp /etc/init.d/skeleton /etc/init.d/launcher

- Puis on édite ce startscript :

gedit /etc/init.d/launcher

Là, plein de choses s'affichent, mais seulement quelques-unes sont importantes :

PATH=/usr/sbin:/usr/bin:/sbin:/bin

DESC="Description du service"

NAME=nomdudemon


DAEMON=/usr/bin/\$NAME

DAEMON_ARGS="--options args"

PIDFILE=/var/run/\$NAME.pid

SCRIPTNAME=/etc/init.d/\$NAME





PATH : il ne faut pas toucher à cette ligne, c'est la liste des PATH

DESC : mettez une courte description de votre launcher.

NAME : mettez le nom de votre exécutable (ici, launcher).

DAEMON : on n'y touche pas (c'est là que se situe votre script).

DAEMON_ARGS : les options de lancement (quand vous lancez la commande, il est possible que vous ayez à mettre des paramètres).

PIDFILE : on laisse.

SCRIPTNAME : on laisse.

Vous remplissez avec vos paramètres comme ci-dessus.

Un exemple avec un script « launcher » dans /usr/bin :

PATH=/usr/sbin:/usr/bin:/sbin:/bin

DESC="Un launcher de mon serveur"

NAME=launcher

DAEMON=/usr/bin/\$NAME

DAEMON_ARGS="-option valeur"

PIDFILE=/var/run/\$NAME.pid

SCRIPTNAME=/etc/init.d/\$NAME



- Nous allons maintenant passer à « l'enregistrement » de notre demon.
chmod +x /etc/init.d/launcher

Update-rc.d

Le programme qui va gérer tous les demons (et bien plus) est « update-rc.d ».
Il crée plusieurs liens depuis /etc/rc0.d/launcher vers /etc/init.d/launcher.
Maintenant, il faut « enregistrer » votre script pour qu'il soit pris en compte.

update-rc.d launcher defaults

update-rc.d pour mettre à jour,
launcher pour le nom de votre script,
defaults options par défaut : placement en bout de file d'attente, pour éviter les conflits
Normalement update-rc.d vous répond (il se peut que le message diffère) :

Adding system startup for /etc/init.d/launcher ...

/etc/rc0.d/K20launcher -> ../init.d/launcher

/etc/rc1.d/K20launcher -> ../init.d/launcher

/etc/rc6.d/K20launcher -> ../init.d/launcher

/etc/rc2.d/S20launcher -> ../init.d/launcher

/etc/rc3.d/S20launcher -> ../init.d/launcher

/etc/rc4.d/S20launcher -> ../init.d/launcher

/etc/rc5.d/S20launcher -> ../init.d/launcher

Vous pouvez maintenant exécuter votre script avec `/etc/init.d/launcher start` ou
`/etc/init.d/launcher stop`.



Supprimer le demon

Si jamais votre script ne fonctionne pas, ou que vous voulez tout simplement enlever votre serveur, il faut exécuter une suite de commandes :

```
/etc/init.d/launcher stop  
update-rc.d -f launcher remove  
rm /etc/init.d/launcher  
rm /usr/bin/launcher
```

/etc/init.d/launcher stop → Termine le programme.

update-rc.d -f launcher remove → Supprime l'enregistrement du script.

rm /etc/init.d/launcher pour supprimer le skeleton modifié.

rm /usr/bin/launcher pour supprimer la copie de votre script.

Votre init.d est maintenant nettoyé. :)





06


LES JOURNAUX DES SERVICES


Les journaux des services et daemon Linux



LES JOURNAUX DES SERVICES

L'exécution des daemons est enregistré dans les journaux systèmes. Lorsque ce dernier ne s'exécute pas correctement, il faut consulter les journaux pour obtenir des informations.





Pour avoir la fin du journal systemd, il faut utiliser la commande
journalctl suivante :

journalctl -xe

```
-- Subject: L'unité (unit) mariadb.service a commencé à démarrer
-- Defined-By: systemd
-- Support: https://www.debian.org/support
--
-- L'unité (unit) mariadb.service a commencé à démarrer.
févr. 04 12:26:36 logipolweb.fr mysqld[20771]: 2019-02-04 12:26:36 140284482265536 [Note] /usr/sbin/mysqld (mysqld 10.1.37-MariaDB-0+deb9u1) starting as process 20771 ...
févr. 04 12:26:36 logipolweb.fr mysqld[20771]: 2019-02-04 12:26:36 140284482265536 [Warning] An old style --language or --lc-message-dir value with language specific part detected: /usr/share/mysql/english/
févr. 04 12:26:36 logipolweb.fr mysqld[20771]: 2019-02-04 12:26:36 140284482265536 [Warning] Use --lc-messages-dir without language specific part instead.
févr. 04 12:26:36 logipolweb.fr mysqld[20771]: 2019-02-04 12:26:36 140284482265536 [Warning] Can't create test file /home/mysqlDatabases/10.1.37-test, lower-test
févr. 04 12:26:36 logipolweb.fr mysqld[20771]: [96B blob data]
févr. 04 12:26:36 logipolweb.fr mysqld[20771]: 2019-02-04 12:26:36 140284482265536 [ERROR] Aborting
févr. 04 12:26:36 logipolweb.fr systemd[1]: mariadb.service: Main process exited, code=exited, status=1/FAILURE
févr. 04 12:26:36 logipolweb.fr systemd[1]: Failed to start MariaDB 10.1.37 database server.
```

la commande journalctl permet à l'administrateur d'interroger et consulter les journaux.

Elle permet aussi de manipuler les journaux, comme par exemple vider les journaux.

journalctl est donc important à connaître pour le débogage du système.



Si vous ne voulez pas que les journaux soient affichés avec less,
utilisez **l'option -no-pager** :

journalctl --no-pager

Afficher les journaux en inversé

Comme vous l'avez remarqué, les journaux sont montrés dans
l'ordre chronologique. Cela signifie que les journaux stockés les
plus anciens sont affichés en premier.

Si vous souhaitez d'abord voir les journaux récents, vous pouvez
afficher les journaux de journal dans l'ordre inverse

avec **l'option -r**:

journalctl -r

Lire le journal de démarrage du système linux

Pour afficher les journaux du dernier démarrage de Linux (boot) :

journalctl -b





Afficher les journaux d'un daemon

Vous pouvez filtrer les journaux pour n'afficher que les logs
d'un daemon et service spécifique.

journalctl -u nomservice

Au besoin pour lister les units :

systemctl list-dependencies

Visualiser les journaux du noyau Linux

Pour afficher les journaux du noyau Linux (kernel), équivaut à la
commande dmesg :

journalctl -k



Afficher les erreurs dans les journaux

Afficher les erreurs dans les journaux où -p 3 signifie priorité ERR, -X fournit des informations de message supplémentaires et -b signifie depuis le dernier démarrage.

journalctl -p 3 -xb

Il s'agit en fait de filtrer sur les niveaux de priorité :

Priorité	Code
0	emerg
1	alert
2	crit
3	err
4	warning
5	notice
6	info
7	debug



Afficher les journaux d'un service/daemon en particulier

Pour afficher les erreurs des journaux pour un service en particulier, utilisez l'option -u :

journalctl -u service.service

Visualiser l'espace disque utilisé par les journaux

journalctl --disk-usage

Fixer la taille et le nombre de journaux

Vous pouvez limiter le nombre de fichiers journaux d'archive. Disons que vous voulez avoir seulement cinq fichiers journaux. Il supprimera les fichiers journaux d'archive les plus âgés ne laissant que le nombre spécifié de fichiers journaux.

journalctl --vacuum-files=5



RESOURCES

LIENS

- <https://www.malekal.com/systemd-service-linux-configuration-et-fonctionnement-daemon/>
- <https://www.malekal.com/comment-fonctionnement-les-services-init-d-linux/>
- <https://www.malekal.com/comment-utiliser-journalctl-pour-voir-lire-journaux-linux-systemd/>
- <https://www.malekal.com/comment-demarrer-arreter-redemarrer-un-service-sur-linux/>
- <https://docs.google.com/document/d/1r-kdUxtmnWQPyfVUdtluXuKqr89lZspj/edit>
- <https://www.linuxtricks.fr/wiki/systemd-les-commandes-essentielles>
- <https://www.linuxtricks.fr/wiki/systemd-creer-des-services-timers-unites>
- <http://sdz.tdct.org/sdz/faire-un-demon-sous-linux.html>





MERCI POUR VOTRE ATTENTION