Chapitre VI : Le processus de démarrage (part 1)

Eric Leclercq

Département IEM
http://ufrsciencestech.u-bourgogne.fr
http://ludique.u-bourgogne.fr/~leclercq

5 mars 2018

Plan du chapitre

Aperçu

- La plupart des systèmes Unix disposent de plusieurs modes de fonctionnement : les niveaux d'exécution
- Un niveau d'exécution = un certain nombre de services démarrés
- Le lancement et l'arrêt des services peut se faire de manière groupée en changeant de niveau d'exécution
- Il existe 7 niveaux d'exécution (en général)
- Seulement trois sont bien définis

Niveaux

- Le niveau 0 correspond à l'arrêt du système : aucun service n'est disponible (à part le redémarrage)
- Le niveau 6 correspond au redémarrage de la machine $\Rightarrow$ le fait de passer dans le niveau d'exécution 6 revient donc à arrêter et à redémarrer la machine
- Le niveau d'exécution 1 correspond au mode de fonctionnement mono utilisateur (*single user*)
- La signification des autres niveaux d'exécution dépend de la distribution en général :
  - le niveau 2 correspond au mode multi-utilisateur avec réseau mais sans X11
  - les niveaux 3 et 4 correspondent au mode multi-utilisateur avec login X11
  - les autres niveaux restent à disposition

Le processus init

- Le programme en charge de gérer les niveaux d'exécution est le processus `init`
- Ce programme est le premier programme lancé par le noyau
- `init` ne peut pas être détruit ou arrêté : c'est le processus père de tous les autres dans le système
- Les rôles fondamentaux d'init sont :
  - de gérer les changements de niveau d'exécution
  - de lancer et arrêter les services du système en fonction de ces niveaux
- `init` s'occupe également de tâches de base concernant la gestion des autres processus

Les zombies

- `init` permet dans certains cas de supprimer les processus zombies
- Un processus zombie est un processus qui vient de se terminer et dont aucun autre processus n'a lu le code de retour
- Les zombies apparaissent généralement lorsque leur processus père se termine avant eux
- En génral c'est le processus père qui lit le code de retour de ses processus fils

Utilisation de Init

La syntaxe suivante est utilisée pour changer le niveau d'exécution :

- `init niveau` ou `telinit niveau`
- niveau est le niveau d'exécution à atteindre
- Il est assez rare d'utiliser la commande directement en général on n'a pas besoin de changer de niveau d'exécution (sauf en mode single)
- Le niveau d'exécution dans lequel le système doit se placer lors de son démarrage peut être précisé en paramètre au noyau lors du boot
  - avec LILO : `boot:linux niveau` ou `boot: linux single`
  - avec GRUB : `kernel /boot/vmlinuz 1`
  - avec Solaris : `boot -s`
  - pour passer en mode monoutilisateur (c'est-à-dire maintenance/single), il suffit de taper la commande au prompt du chargeur de noyau

Le fichier inittab

- Le comportement d'`init` est défini dans le fichier de configuration `/etc/inittab`
- Ce fichier contient :
  - la description des niveaux d'exécution
  - le niveau par défaut dans lequel le système se place au démarrage
  - les actions qu'`init` doit effectuer lorsque certains événements arrivent
- Il est indiqué quels sont les scripts qui doivent être exécutés lors du changement de niveau d'exécution
- Il est fortement déconseillé de toucher au fichier `/etc/inittab`

Structure du fichier

```
#
# inittab       This file describes how the INIT process should set up
#               the system in a certain run-level.
#
# Author:       Miquel van Smoorenburg, <miquels@drinkel.nl.mugnet.org>
#               Modified for RHS Linux by Marc Ewing and Donnie Barnes
#

# Default runlevel. The runlevels used by RHS are:
#   0 - halt (Do NOT set initdefault to this)
#   1 - Single user mode
#   2 - Multiuser, without NFS (The same as 3, if you do not have netwo
#   3 - Full multiuser mode
#   4 - unused
#   5 - X11
#   6 - reboot (Do NOT set initdefault to this)
#
id:5:initdefault:
```

Structure du fichier

```
# System initialization.
si::sysinit:/etc/rc.d/rc.sysinit

l0:0:wait:/etc/rc.d/rc 0
l1:1:wait:/etc/rc.d/rc 1
l2:2:wait:/etc/rc.d/rc 2
l3:3:wait:/etc/rc.d/rc 3
l4:4:wait:/etc/rc.d/rc 4
l5:5:wait:/etc/rc.d/rc 5
l6:6:wait:/etc/rc.d/rc 6

# Trap CTRL-ALT-DELETE
ca::ctrlaltdel:/sbin/shutdown -t3 -r now
```

Structure du fichier

```
# When our UPS tells us power has failed, assume we have a few minutes
# of power left.  Schedule a shutdown for 2 minutes from now.
# This does, of course, assume you have powerd installed and your
# UPS connected and working correctly.
pf::powerfail:/sbin/shutdown -f -h +2 "Power Failure; System Shutting Down"

# If power was restored before the shutdown kicked in, cancel it.
pr:12345:powerokwait:/sbin/shutdown -c "Power Restored; Shutdown Cancelled"

# Run gettys in standard runlevels
1:2345:respawn:/sbin/mingetty tty1
2:2345:respawn:/sbin/mingetty tty2
3:2345:respawn:/sbin/mingetty tty3
4:2345:respawn:/sbin/mingetty tty4
5:2345:respawn:/sbin/mingetty tty5
6:2345:respawn:/sbin/mingetty tty6

# Run xdm in runlevel 5
x:5:respawn:/etc/X11/prefdm -nodaemon
```

Structure du fichier (Solaris)

```
ap::sysinit:/sbin/autopush -f /etc/iu.ap
ap::sysinit:/sbin/soconfig -f /etc/sock2path
fs::sysinit:/sbin/rcS sysinit           >/dev/msglog 2<>/dev/msglog </dev/console
is:3:initdefault:
p3:s1234:powerfail:/usr/sbin/shutdown -y -i5 -g0 >/dev/msglog 2<>/dev/msglog
sS:s:wait:/sbin/rcS                     >/dev/msglog 2<>/dev/msglog </dev/console
s0:0:wait:/sbin/rc0                     >/dev/msglog 2<>/dev/msglog </dev/console
s1:1:respawn:/sbin/rc1                  >/dev/msglog 2<>/dev/msglog </dev/console
s2:23:wait:/sbin/rc2                    >/dev/msglog 2<>/dev/msglog </dev/console
s3:3:wait:/sbin/rc3                     >/dev/msglog 2<>/dev/msglog </dev/console
s5:5:wait:/sbin/rc5                     >/dev/msglog 2<>/dev/msglog </dev/console
s6:6:wait:/sbin/rc6                     >/dev/msglog 2<>/dev/msglog </dev/console
fw:0:wait:/sbin/uadmin 2 0              >/dev/msglog 2<>/dev/msglog </dev/console
of:5:wait:/sbin/uadmin 2 6              >/dev/msglog 2<>/dev/msglog </dev/console
rb:6:wait:/sbin/uadmin 2 1              >/dev/msglog 2<>/dev/msglog </dev/console
sc:234:respawn:/usr/lib/saf/sac -t 300
co:234:respawn:/usr/lib/saf/ttymon -g -h -p "‘uname -n‘ console login: " -T sun -d /dev/console -l console
```

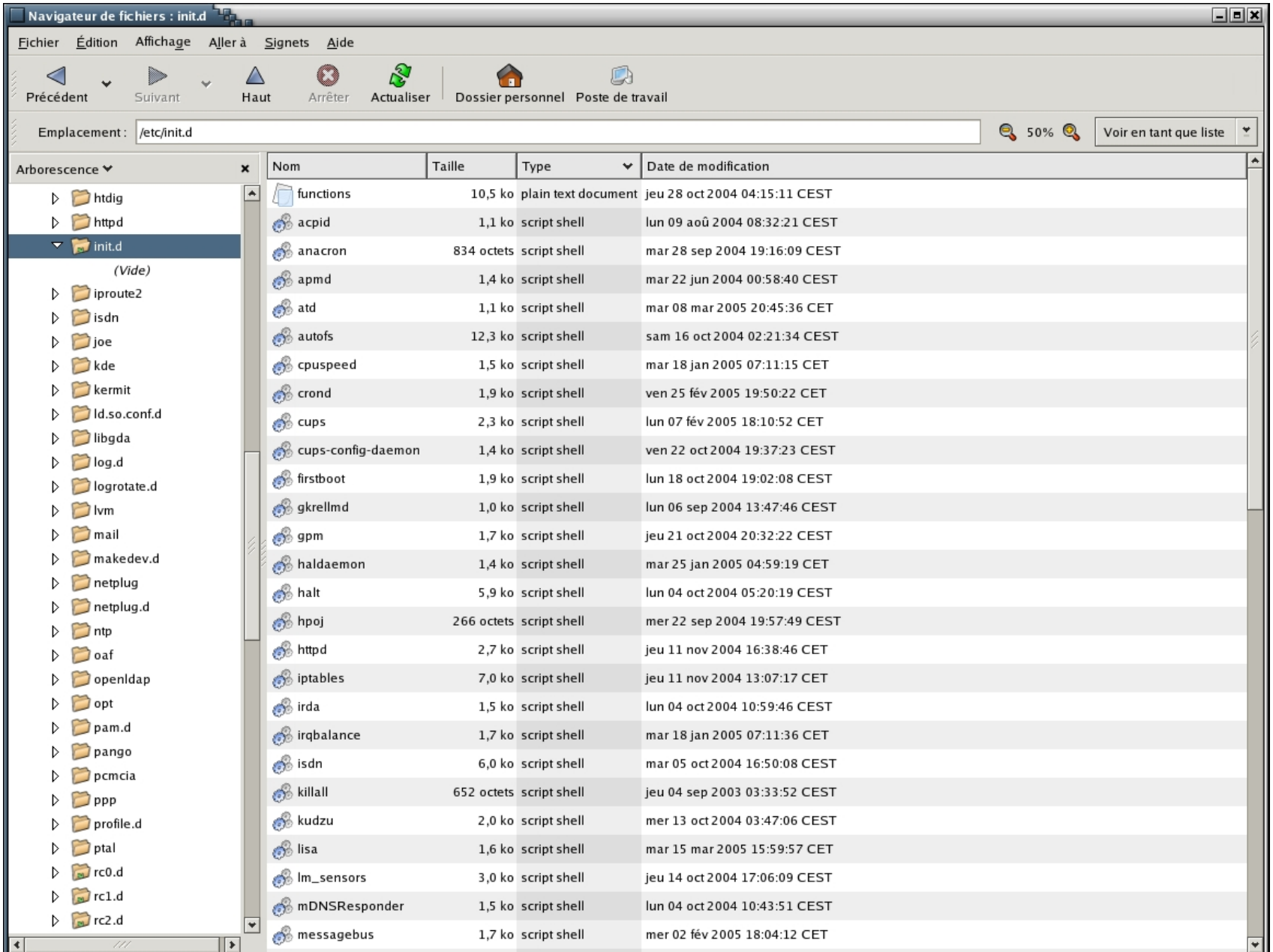
Enchaînement des scripts

- Lorsqu'on change de niveau d'exécution, `init` appelle les scripts de configuration pour effectuer les opérations de configuration, de lancement et d'arrêt des différents services
- Ces scripts sont spécifiés dans le fichier `/etc/inittab`
- En général ils sont placés dans le répertoire `/etc/rc.d/` ou `/sbin/init.d/`. Ce répertoire contient donc (en général) :
  - le script exécuté lors du démarrage du système
  - les scripts exécutés lors de l'entrée dans un niveau d'exécution
  - les scripts exécutés lors de la sortie d'un niveau d'exécution
- On préfère une organisation plus rigoureuse ne place dans `init.d` que les scripts des services

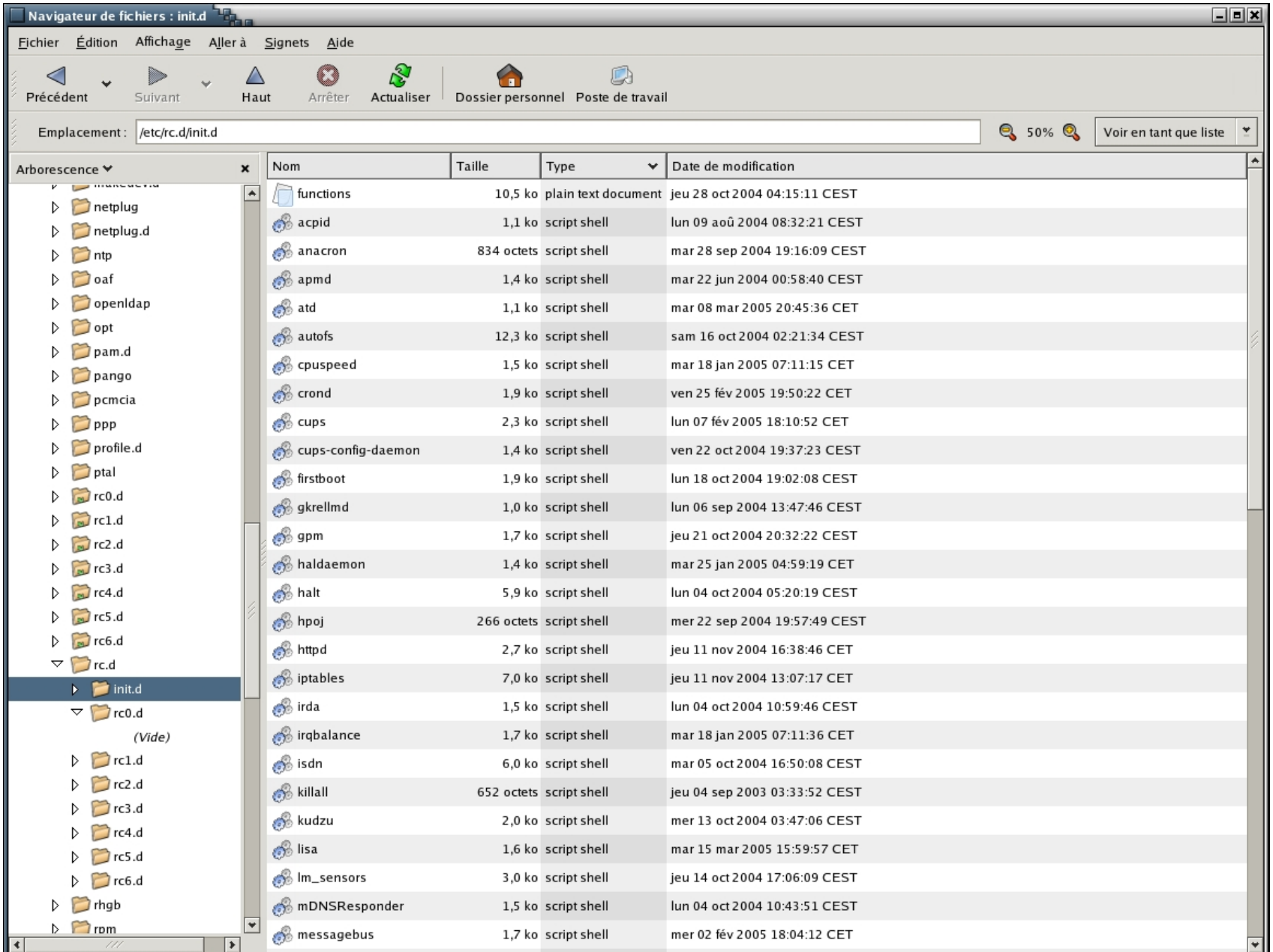
Enchaînement des scripts

- Le script appelé lors du démarrage du niveau est en général spécifié directement dans `/etc/inittab`
- Vous pouvez y ajouter les commandes spécifiques à votre système (par exemple les commandes de configuration du matériel)
- Ce fichier n'est exécuté qu'une seule fois et est placé directement dans `/etc/rc.d/` ou dans `/sbin/init.d/`
- Les scripts appelés lors du changement de niveau d'exécution sont souvent placés dans des sous-répertoires du répertoire rc.d ou init.d.
- Ils sont classés à raison d'un répertoire par niveau d'exécution.
- Ces sous-répertoires portent le nom de rc0.d, rc1.d, rc2.d, etc.

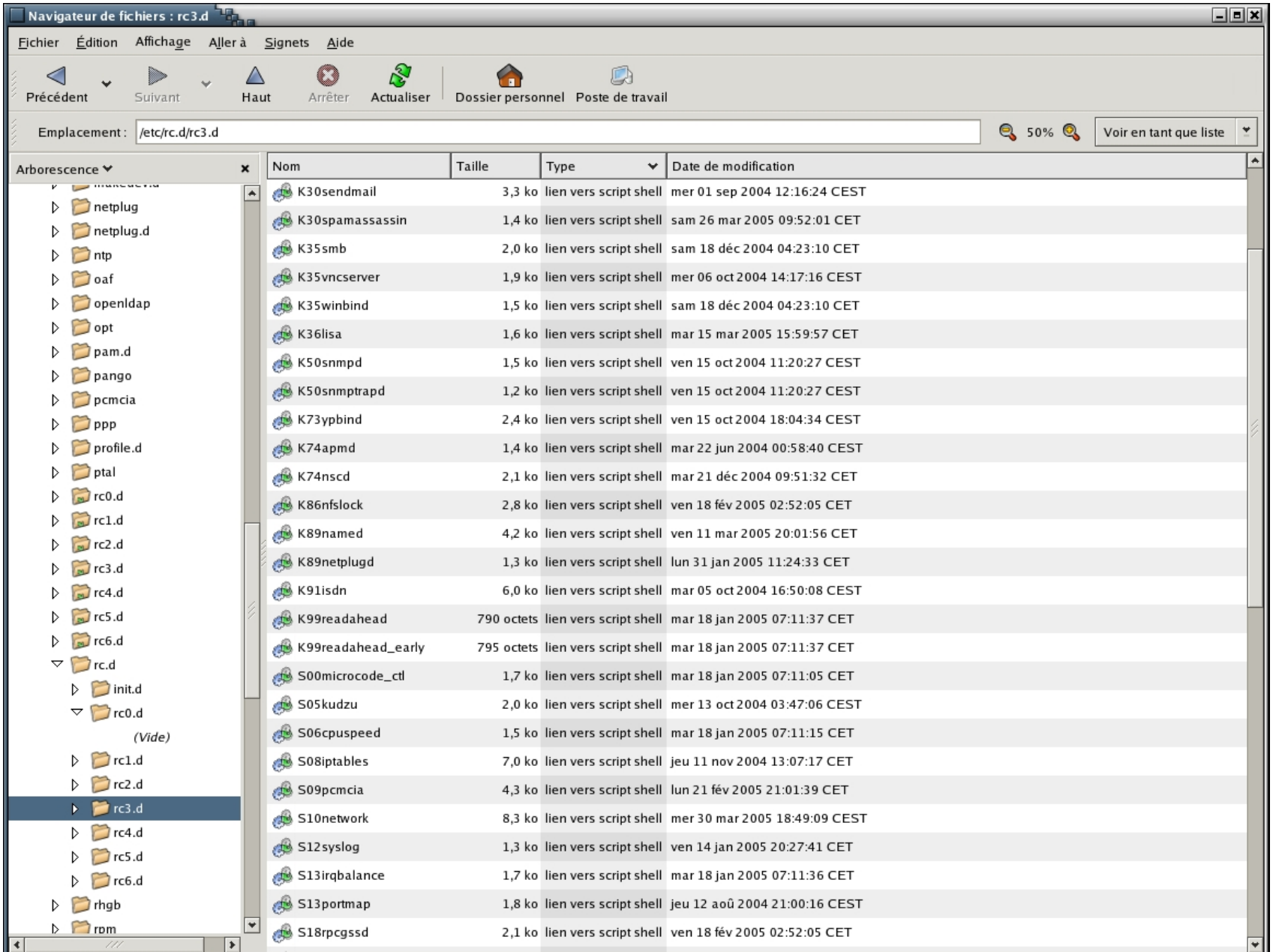
Enchaînement des scripts



Enchaînement des scripts



Enchaînement des scripts



Enchaînement des scripts

- un seul script est exécuté par `init` lorsqu'on change de niveau d'exécution
- il se charge d'exécuter les bons scripts dans les sous-répertoires de `rc.d` ou `init.d`.
- ce script principal est appelé avec le numéro du niveau d'exécution en paramètre, et il commence par appeler les scripts de sortie du niveau d'exécution courant, puis les scripts d'entrée dans le nouveau niveau d'exécution
- la distinction entre les scripts d'entrée et de sortie dans chaque répertoire rcn.d se fait par la première lettre du script. Sur certaines distributions, la lettre K correspond aux scripts de sortie et la lettre S au script d'entrée
- l'ordre dans lequel ces scripts doivent être exécutés est indiqué par le nombre

Enchaînement des scripts

- Il est assez courant (et même recommandé) que les répertoires `rcX.d` ne contiennent que des liens symboliques vers les fichiers de scripts ;
- Ceux-ci sont placés directement dans le répertoire `/etc/rc.d/` ou plus souvent dans `/etc/init.d`
- Un même script peut être utilisé pour différents niveaux d'exécution ;
- Il est assez courant qu'un scripts gère à la fois l'entrée et la sortie du niveau d'exécution selon le paramètre qu'il reçoit lors de son appel
- Parmi les paramètres, on retrouve :
  - start, pour le démarrage du service
  - stop, pour son arrêt

Enchaînement des scripts

- De cette manière, vous pouvez ajouter ou supprimer des services simplement dans chaque niveau d'exécution.
- Il suffit d'écrire un fichier script capable de prendre en paramètre l'action à réaliser sur le service (start ou stop), de le placer dans /etc/rc.d/ ou /etc/init.d/) et de créer les liens dans les sous-répertoires /etc/rc.d/rc?.d/
- Dès lors, votre service sera arrêté ou redémarré selon le niveau d'exécution dans lequel passera le système.