

MODULE 2

Gestion des services

Objectifs de ce module :

- ✓ *Arrêter et démarrer des services*
- ✓ *Ajouter ou enlever des services au démarrage*
- ✓ *Connaître la structure des fichiers de service*
- ✓ *Utiliser les principales commandes de gestion des services.*

Table des matières

Module 2 - Gestion des services

| <i>Sujets</i> | <i>Page</i> |
|--|--------------------|
| Introduction..... | 3 |
| La séquence de démarrage..... | 3 |
| Le fichier /etc/fstab..... | 4 |
| 1 Qu'est-ce que systemd ?..... | 5 |
| 2 Qu'est-ce qu'un service ?..... | 5 |
| 3 Historique..... | 5 |
| 4 Remarques..... | 5 |
| 5 Quand sont lancés les services ?..... | 6 |
| 6 Pourquoi me préoccuper des services ?..... | 6 |
| 7 Niveaux d'exécution / cibles..... | 7 |
| 8 Les outils graphiques..... | 7 |
| 9 Quelques informations sur systemd..... | 8 |
| 9.1 Les grands principes à la base de systemd..... | 8 |
| 9.2 Notion d'unité..... | 8 |
| 9.3 Essai de typologie des services..... | 8 |
| 9.3.1 Type de service..... | 8 |
| 9.3.2 Allons plus loin avec les services de type "oneshot"..... | 9 |
| 9.4 Emplacement des fichiers de configuration..... | 10 |
| 9.5 Analyse du temps de démarrage du système..... | 10 |
| 10 Gestion des services..... | 11 |
| 10.1 Changer le niveau / la cible d'exécution..... | 11 |
| 10.1.1 Changer le niveau / la cible d'exécution immédiatement..... | 11 |
| 10.1.2 Changer le niveau / la cible d'exécution par défaut..... | 12 |
| 10.2 Lancer un service..... | 12 |
| 10.3 Arrêter un service..... | 13 |
| 10.4 Activer un service..... | 13 |
| 10.5 Désactiver un service..... | 14 |
| 10.6 Vérifier l'état d'un service..... | 14 |
| 10.7 Quand utiliser les commandes SysVinit / UpStart ? Quand utiliser les commandes systemd ?..... | 15 |
| 11 États d'un service..... | 15 |
| 11.1 États d'un service avec SysVinit / Upstart..... | 15 |
| 11.2 États d'un service avec systemd..... | 15 |

Introduction

Les services permettent de répondre à des requêtes du système face aux demandes des usagers. Ces services utilisent des ports du système et il est donc par conséquent extrêmement important de connaître le numéro des ports utilisés et les services associés à ces ports pour pouvoir gérer adéquatement la sécurité de votre système. Il se peut que vous vouliez rendre accessible un service en particulier (comme un serveur Web par exemple). Il faut donc connaître comment activer ou désactiver de tels services afin de minimiser les problèmes potentiels et aussi de diminuer la charge du serveur en activant des services dont on ne se servira pas.

Il existe plusieurs moyens de gérer l'accès aux services. Nous verrons dans ce module, une manière graphique de le réaliser, les commandes consoles ainsi qu'un utilitaire semi-graphique (ntsysv).

La façon la plus simple de refuser l'accès à un service est tout simplement de le désactiver. Il existe principalement une commande pour gérer les services sous un système Fedora, il s'agit de la commande "systemctl".

La séquence de démarrage

Cette séquence comporte plusieurs étapes dont voici les principales:

1. Du BIOS jusqu'au Noyau
 - Le BIOS s'initialise.
 - Le BIOS charge le MBR (Le Master Boot Record).
 - (Assumant que GRUB est installé dans le MBR) Le chargeur initial localise et charge le chargeur secondaire.
 - GRUB présente ensuite le menu de démarrage, c'est-à-dire, le choix du système d'exploitation à démarrer s'il y a plus d'un système d'installé.
 - GRUB charge ensuite le noyau sélectionné et ensuite le ramdisk si celui-ci est présent.
 - GRUB se termine et le noyau prend charge du reste de la séquence.
 - Le noyau se décompresse, se charge et active les modules au démarrage.
 - Le premier processus est alors lancé, il s'agit de "init".
 - C'est "init" qui continue ensuite le processus de démarrage.

Le fichier /etc/fstab

C'est dans ce fichier que sont listés les différents systèmes de fichiers disponibles sur votre système. Pour chacun d'eux, on trouvera un nom, un point de montage, un type (ext3 ou ntfs par exemple) puis des options de montage à utiliser.

Le fichier est un fichier de configuration Linux typique: fichier texte standard, avec une entrée par ligne. Chaque entrée contient plusieurs champs séparés par un ou des espaces ou tabulations. Les champs n'ont pas besoin d'être alignés d'une entrée à l'autre. Les lignes vides ou commençant par # sont ignorées.

Par exemple, on y verra quelque chose comme:

```
#
# /etc/fstab
# Created by anaconda on Mon Sep 30 15:04:00 2013
#
# Accessible filesystems, by reference, are maintained under '/dev/disk'
# See man pages fstab(5), findfs(8), mount(8) and/or blkid(8) for more info
#
UUID=5308f1c3-f0e6-4c52-9084-3f14d6497bca /          ext4 defaults        1 1
UUID=8019c2c7-b0ff-4a2f-9cb3-2c22b2075f4c sw         swap defaults         0 0
tmpfs          /dev/shm      tmpfs defaults         0 0
devpts         /dev/pts      devpts gid=5,mode=620      0 0
sysfs         /sys          sysfs defaults         0 0
proc          /proc         proc  defaults         0 0
```

Dans ce fichier, les champs, dans l'ordre, sont:

- Nom de la partition (ça peut être un numéro compliqué si c'est le système qui l'a baptisé, mais ça n'a aucune importance)
- Point de montage
- Type de système de fichiers (ext4 pour Linux, NTFS pour Windows, swap et proc pour des partitions systèmes spéciales, et plein d'autres possibilités encore)
- Options: une liste d'options à utiliser lors du montage, séparées par des virgules (et sans espaces!). Les options les plus courantes:
 - auto ou noauto: détermine si la partition est montée automatiquement ou non au démarrage
 - user ou nouser: détermine si les usagers "ordinaires" peuvent monter la partition ou si seulement root peut le faire
 - exec ou noexec: détermine s'il sera possible ou non d'exécuter les programmes se trouvant sur cette partition. Noexec est pratique pour les partitions Windows
 - ro: read only
 - rw: read and write
 - sync ou async: détermine si l'écriture physique sera synchronisée ou non avec l'écriture logique.
 - defaults: utilise les options par défaut, qui sont rw,suid,dev,exec,auto,nouser,async

- Dump: dump est un logiciel de prise de backups. Si un 0 est placé dans cette colonne, la partition sera ignorée par dump.
- Fsck: fsck (file system check) est un logiciel de vérification d'intégrité d'un système de fichiers. 0 dans cette colonne signifie que cette partition ne sera jamais vérifiée par fsck. Les partitions ayant autre chose que 0 seront vérifiées dans l'ordre donné par cette valeur, du plus petit au plus grand.

Ce fichier est donc essentiel au bon fonctionnement du système. Si un jour vous faites des modifications aux disques durs ou partitions, il faut vous assurer que /etc/fstab est mis à jour correctement.

1 Qu'est-ce que systemd ?

systemd est un gestionnaire de service. Il permet de les lancer, les arrêter, les relancer, d'en assurer un suivi, etc.

C'est aussi un gestionnaire de session. Il permet l'utilisation du même matériel en passant d'une session (~logging) à une autre sans que cela pose de problème.

Il permet encore de gérer les points de montage qu'ils soient automatiques ou non.

systemd intègre et va certainement intégrer d'autres aspects. Le présent tutoriel s'intéresse principalement à l'aspect service, qui est le cœur de systemd.

2 Qu'est-ce qu'un service ?

Un service est un programme qui est exécuté en tâche de fond. C'est à dire qu'il effectue son travail sans interaction directe avec l'utilisateur.

Certains services sont des composants essentiels du système.

L'autre petit nom d'un service est *daemon*.

3 Historique

Différents systèmes de gestion des services sont utilisés par les distributions GNU/Linux. Par exemple :

- SysVinit (init System V).
- UpStart.
- systemd.

SysVinit est le classique système de démarrage et de gestion des services qui a été utilisé jusqu'à Fedora 8.

UpStart a remplacé SysVinit dans Fedora 9, jusqu'à Fedora 14.

systemd prend place dans Fedora à partir de la version 15.

4 Remarques

- L'ensemble des commandes que nous verrons seront données pour SysVinit / Upstart et pour systemd.

SysVinit et Upstart partagent le même jeu de commande qui est présenté pour faire le lien avec les anciennes pratiques et systemd.

systemd introduit un nouveau jeu de commande, tout en permettant que les anciennes commandes soient partiellement compatibles.

- Les unités de systemd.

systemd dispose d'une notion d'unité. Les services sont un type d'unité.

Ce tutoriel est clairement orienté autour de la gestion des services.

- Nous ne rentrerons pas dans les détails de l'ensemble des possibilités de systemd.

Quelques informations supplémentaires seront donnés sur systemd quand cela est nécessaire à la bonne compréhension de la gestion des services.

Pour ceux et celles qui voudront aller plus loin, nous donnerons quelques pointeurs à la fin du tutoriel.

5 Quand sont lancés les services ?

Certains services sont lancés par défaut au démarrage du système.

Par défaut, il s'agit d'un choix des concepteurs de la distribution. Ce choix, comprenant des services essentiels et d'autres qui le sont moins.

Vous pouvez choisir de désactiver certains services qui ne sont pas essentiels.

Pourquoi ? Parce que vous n'en avez pas besoin. Ou parce que vous n'en avez pas besoin souvent, voire très rarement ou même jamais.

Vous pouvez aussi choisir d'arrêter momentanément des services. Par exemple, pour voir ce que ça fait (ou ne fait plus ...), avant de les désactiver complètement.

Vous pouvez activer des services qui ne sont pas par défaut, mais dont, vous, vous avez besoin.

Et enfin, vous pouvez lancer certain service uniquement au besoin.

6 Pourquoi me préoccuper des services ?

Les principales raisons sont :

- La sécurité.
- Optimiser le lancement du système.
- Optimiser le système.

La sécurité ?

Certain services sont à "l'écoute" de sollicitations externes. Toutes sollicitations externes peut venir d'une personne malveillante. Et donc, moins il y a de services de ce type en exécution, moins il y a de portes d'entrées potentielles pour ces personnes.

Optimiser le lancement du système ?

Plus il y a de services qui sont lancés au démarrage, plus cela prend du temps pour que le système deviennent fonctionnel.

Il s'agit d'ailleurs d'une des principales motivation de la naissance de Upstart et systemd.

Optimiser le système ?

Les concepteurs d'une distribution vont, généralement, essayer de faire en sorte que la distribution par défaut puisse convenir au plus grand nombre. Par exemple, pour cela, il faut être capable de gérer différents matériels. Ces matériels, peuvent être présents ou pas sur la machine de chaque utilisateur. Et le service associé, peut être utile ou pas.

Cette utilité, vous êtes finalement le seul ou la seule à pouvoir la définir. Certain services sont donc présent pour répondre au besoin du plus grand nombre, mais ne vous sont pas forcément utile. Et au contraire, des services peuvent vous être utile, bien qu'ils ne sont pas lancés par défaut.

7 Niveaux d'exécution / cibles

SysVinit et Upstart ont une notion de niveau d'exécution (RunLevel).

systemd a une notion de cible (target).

Sans entrer plus dans les détails, à un niveau d'exécution ou une cible correspond un mode de fonctionnement du système où certains services sont arrêtés tandis que d'autres sont lancés.

Le tableau ci-dessous décrits les différents modes de fonctionnement standard.

| Les modes de fonctionnements | | |
|--|--|---|
| Niveaux d'exécution SysVinit / Upstart | Cibles systemd | Commentaires |
| 0 | runlevel0.target, poweroff.target | Arrêt du système |
| 1,s, single | runlevel1.target, rescue.target | Mode utilisateur unique / mode maintenance (Single User Mode) |
| 3 | runlevel3.target, multi-user.target | Mode multi-utilisateur non graphique |
| 2, 4 | runlevel2.target, runlevel4.target, multi-user.target | Ces modes sont identiques au mode multi-utilisateur non graphique. Ces modes peuvent être adaptés aux besoins |
| 5 | runlevel5.target, graphical.target | Mode multi-utilisateur graphique. Il s'agit du mode par défaut de Fedora |
| 6 | runlevel6.target, reboot.target | Redémarrage du système |
| emergency | emergency.target | Shell d'urgence |

8 Les outils graphiques

- system-config-services

Cette commande permet de lancer / arrêter et activer / désactiver les services, tout en visualisant leurs états. Elle permet aussi de changer le mode de fonctionnement.

Ces points sont surtout vrai pour SysVinit et Upstart. Elle peut toujours être utilisée avec systemd, mais certaines fonctionnalités ne sont plus accessibles.

L'installation de cette commande se fait par `yum install system-config-services`.

- `systemadm`

Cette commande est propre à `systemd`. Elle permet les mêmes actions que `system-config-services`, et d'autres spécifiques à `systemd` comme de visualiser les dépendances, les cibles, et d'autres types d'unités que les services.

Avant Fedora 18, l'installation de cette commande se fait par `yum install systemd-gtk`.

A partir de Fedora 18, l'installation de cette commande se fait par `yum install systemd-ui`.

Ces outils graphiques ne seront pas plus explicités. Ils sont suffisamment intuitifs.

9 Quelques informations sur `systemd`

9.1 Les grands principes à la base de `systemd`

- Le grand principe de `systemd` est de créer les canaux de communication entre les processus avant de lancer les processus des services.
- La synchronisation effectuée par le système permet de gérer une partie des dépendances.

Par exemple, si un service utilise une socket pour recevoir les requêtes à traiter, si la socket est créée par `systemd` les clients de ce services peuvent accéder à la socket avant que le service soit effectivement lancé et donc y écrivent leurs requêtes sans être en erreur par absence de la socket.

- L'autre mécanisme de communication inter-processus est DBUS. DBUS utilisant des sockets ...

9.2 Notion d'unité

- `systemd` utilise une notion d'unité ("Unit"). Il y a différentes type d'unités : `automount`, `path`, `mount`, `service`, `socket`, `target`, `timer`, `wants`.

Pour notre explication sur les services, nous retenons les unités de type `socket` et `service`.

- Les unités de type `socket` permettent à `systemd` de connaître les socket à créer.
- Les unités de type `service` permettent à `systemd` de connaître les services à lancer. Bien entendu, les socket sont traitées par `systemd` avant les services.

9.3 Essai de typologie des services

9.3.1 Type de service

- L'un des paramètres de configuration d'un service avec systemd est "Type=" avec comme valeur possible : simple, forking, oneshot, dbus, notify.
- Un service de type "simple" (type par défaut) lance un processus principal. Dans le cas où ce processus offre une fonctionnalité à d'autres processus sur le système, ses canaux de communication doivent être installés avant que le processus principal soit démarré. Ce qui est rendu possible par l'activation des sockets, et autres canaux de communication, tel qu'indiqué ci-dessus. Ainsi, systemd peut traiter les autres unités sans se préoccuper de la fin du lancement d'une unité de type "simple".
- Un service de type "forking", lance un processus père qui créera un processus fils dans le cadre de son démarrage. Le processus parent est prévu pour s'arrêter une fois le démarrage complet et que tous les canaux de communication sont mis en place. Le processus fils continue à fonctionner en tant que le processus principal du service. systemd poursuit le traitement des autres unités une fois que le processus père se termine.

Ce type de service correspond aux services UNIX traditionnels.

- Un service de type "oneshot" est similaire à un service de type "simple". Cependant, systemd attend que le processus se termine avant de continuer ses traitements.

Ce type de service est typiquement utilisé comme équivalent aux commandes lancées au démarrage via les scripts d'init system V. Cela permet à systemd de remplacer ce mécanisme.

De ce fait, avec systemd des nouveaux services apparaissent, alors qu'ils auraient été simplement des scripts d'init avec SysVinit.

- Un service de type "dbus" est similaire à un service de type "simple". Cependant, le processus du service doit obtenir un nom via D-Bus. systemd pourra alors traiter les autres unités.
- Un service de type "notify" est similaire à un service de type "simple". Cependant, c'est le processus du service qui avertira systemd (via la fonction `sd_notify(3)`) qu'il peut traiter les autres unités.

9.3.2 Allons plus loin avec les services de type "oneshot"

- Différents paramètres peuvent être utilisés dans la configuration des services. Nous allons en présenter quelques uns dans la perspective des services de type "oneshot".

- ExecStart permet d'indiquer la commande à exécuter au lancement du service. Ce paramètre est obligatoire pour tout les types de service.
- ExecStop permet d'indiquer une commande a exécuter pour arrêter le service. Ce paramètre est facultatif.
- RemainAfterExit à la valeur "yes" permet d'indiquer que quand la commande de lancement (ExecStart) est terminée, le service est considéré comme toujours lancé. Ce paramètre est très utile pour les services de type "oneshot" qui exécutent une commande à leur lancement (ExecStart) sans qu'il y ait un processus spécifique qui reste en exécution.

Un exemple est le service iptables. Ci-dessous, je donne un extrait de son fichier de configuration :

```
[Service]
Type=oneshot
RemainAfterExit=yes
ExecStart=/usr/libexec/iptables.init start
ExecStop=/usr/libexec/iptables.init stop
```

Nous pouvons voir que :

- A son lancement, la commande `/usr/libexec/iptables.init start` est exécutée. Cette commande va permettre de charger des modules iptables et les règles iptables que le modules netfilter du noyau linux va prendre en compte.
- A son arrêt, la commande `/usr/libexec/iptables.init stop` est exécutée. Cette commande va permettre de remettre en place la politique de traitements iptables par défaut et de décharger les modules iptables.
- Le service iptables continuera d'apparaître comme actif après la fin de la commande `/usr/libexec/iptables.init start` et jusqu'à ce qu'il soit explicitement arrêté.

Finalement, un service de type "oneshot" configuré avec RemainAfterExit=yes s'apparente bien à un service classique de type UNIX.

Remarquez qu'il y a des services de type oneshot, avec l'option ExecStart et RemainAfterExit sans option ExecStop. C'est juste qu'il n'y a pas de commande déclenchée à l'arrêt du service.

9.4 Emplacement des fichiers de configuration

- En plus des types de service, les fichiers de configuration des services permettent d'indiquer à systemd d'autres dépendances explicites à d'autres unités.
- La configuration des services est effectuées dans le répertoire `/lib/systemd/system` avec des fichiers d'extensions ".service". Il s'agit de l'emplacement standard.
- Il est possible d'utiliser le répertoire `/etc/systemd/system` pour effectuer des

modifications sans risque quelles soient perdues suite à une mise à jour du système

Pour cela, il suffit de créer le fichier `/etc/systemd/system/XXX.service` avec le contenu du fichier `/lib/systemd/system/xxx.service`, puis de modifier à votre convenance ce nouveau fichier.

9.5 Analyse du temps de démarrage du système

- La commande `systemd-analyze blame`, permet de lister le temps de démarrage des différents services. Il est ainsi possible de repérer les services très ou trop long à se lancer.

Par exemple :

```
$ systemd-analyze blame
6207ms udev-settle.service
5228ms
cryptsetup@luks\x2d9899b85d\x2df790\x2d4d2a\x2da650\x2d8b7d2fb92cc3.service
735ms NetworkManager.service
642ms avahi-daemon.service
600ms abrtd.service
517ms rtkit-daemon.service
478ms fedora-storage-init.service
396ms dbus.service
390ms rpcidmapd.service
346ms systemd-tmpfiles-setup.service
322ms fedora-sysinit-unhack.service
316ms cups.service
310ms console-kit-log-system-start.service
309ms libvirtd.service
303ms rpcbind.service
298ms ksmtuned.service
288ms lvm2-monitor.service
281ms rpcgssd.service
277ms sshd.service
276ms livesys.service
267ms iscsid.service
236ms mdmonitor.service
234ms nfslock.service
223ms ksm.service
218ms mcelog.service
...
```

Et vous obtenez la liste des services lancés au démarrage classés de celui qui a pris le plus de temps à celui en a pris le moins.

10 Gestion des services

Dans ce chapitre, nous allons voir comment :

- Changer de niveau / cibles d'exécution.
- Lancer un service.
- Arrêter un service.
- Activer un service.

- Désactiver un service.
- Vérifier l'état d'un service.

tout ceci en ligne de commande.

10.1 Changer le niveau / la cible d'exécution

10.1.1 Changer le niveau / la cible d'exécution immédiatement

Pour SysVinit / Upstart

```
telinit NIVEAU
```

Par exemple :

- `telinit 3`, pour passer en mode multi-utilisateur non graphique.
- `telinit 5`, pour passer en mode multi-utilisateur graphique.

Pour systemd

```
systemctl isolate CIBLE.target
```

Par exemple :

- `systemctl isolate multi-user.target` ou `systemctl isolate runlevel3.target`, pour passer en mode multi-utilisateur non graphique.
- `systemctl isolate graphical.target` ou `systemctl isolate runlevel5.target`, pour passer en mode multi-utilisateur graphique.

10.1.2 Changer le niveau / la cible d'exécution par défaut

Ci-dessous, nous allons voir comment changer le niveau / la cible d'exécution par défaut. C'est à dire ce qui est pris en compte au démarrage du système.

Pour SysVinit / Upstart, il est nécessaire d'éditer le fichier `/etc/inittab` et de modifier la ligne avec le mot clé "initdefault" avec la valeur désirée. Par exemple :

- Remplacer "id:3:initdefault:" par "id:5:initdefault:" pour passer du mode multi-utilisateur non graphique au mode multi-utilisateur graphique.
- Remplacer "id:5:initdefault:" par "id:3:initdefault:" pour passer du mode multi-utilisateur graphique au mode multi-utilisateur non graphique.

Pour systemd

```
systemctl set-default CIBLE.target
```

Par exemple :

- `ln -sf /lib/systemd/system/multi-user.target`

`/etc/systemd/system/default.target`, pour passer en mode multi-utilisateur non graphique par défaut.

- `ln -sf /lib/systemd/system/graphical.target /etc/systemd/system/default.target`, pour passer en mode multi-utilisateur graphique par défaut.

10.2 Lancer un service

Pour SysVinit / Upstart

```
services NOM_SERVICE start
```

Par exemple :

- `services network start`, pour lancer le service network.

Pour systemd

```
systemctl start NOM.service
```

Par exemple :

- `systemctl start network.service`, pour lancer le service network.



Lancement versus activation

Le lancement d'un service n'est pas suffisant pour que celui-ci soit redémarré au prochaine lancement du système. Il est nécessaire que celui-ci soit activé, cf. ci-dessous.

10.3 Arrêter un service

Pour SysVinit / Upstart

```
services NOM_SERVICE stop
```

Par exemple :

- `services network stop`, pour arrêter le service network.

Pour systemd

```
systemctl stop NOM.service
```

Par exemple :

- `systemctl stop network.service`, pour arrêter le service network.



Arrêt versus désactivation

L'arrêt d'un service n'est pas suffisant pour que celui-ci soit pas redémarré au prochaine lancement du système. Il est nécessaire que le service soit désactivé, cf. ci-dessous.

10.4 Activer un service

Pour SysVinit / Upstart

```
chkconfig NOM_SERVICE on
```

Par exemple :

- `chkconfig network on`, pour activer le service network.

Pour systemd

```
systemctl enable NOM.service
```

Par exemple :

- `systemctl enable network.service`, pour activer le service network.



Activation versus lancement

L'activation d'un service permet uniquement de le lancer, par défaut, au démarrage du système. Pour le lancer immédiatement, cf. ci-dessus.

10.5 Désactiver un service

Pour SysVinit / Upstart

```
chkconfig NOM_SERVICE off
```

Par exemple :

- `chkconfig network off`, pour désactiver le service network.

Pour systemd

```
systemctl disable NOM.service
```

Par exemple :

- `systemctl disable network.service`, pour désactiver le service network.



Désactivation versus arrêt

La désactivation d'un service permet uniquement de ne pas le lancer, par défaut, au démarrage du système. Pour l'arrêter immédiatement, cf. ci-dessus.

10.6 Vérifier l'état d'un service

Pour SysVinit / Upstart

```
chkconfig --list NOM_SERVICE
```

et

```
service NOM_SERVICE status
```

Par exemple :

- `chkconfig --list network`, pour vérifier l'état d'activation du service network.
- `service network status`, pour vérifier l'état de lancement du service network.

Pour systemd

```
systemctl is-enabled NOM.service
```

et

```
systemctl status NOM.service
```

Par exemple :

- `systemctl is-enabled network.service`, pour vérifier l'état d'activation du service network.
- `systemctl status network.service`, pour vérifier l'état de lancement du service network.

10.7 Quand utiliser les commandes SysVinit / UpStart ? Quand utiliser les commandes systemd ?

A partir de Fedora 15, les commandes `chkconfig` et `services` sont interchangeables avec la commandes `systemctl`.

Un mécanisme d'appel de la commande adéquate est mis en place. Par exemple, l'utilisation de la commande `service` sur un service géré par `systemd` appellera la commande `systemctl` et inversement.

Il en est de même avec la commande `chklist`, sauf pour l'option `chklist --list` qui ne listera que les services qui ne sont pas gérés par `systemd`.

11 États d'un service

11.1 États d'un service avec SysVinit / Upstart

- La commande `chkconfig --list NOM_SERVICE` permet d'indiquer, pour chaque niveau d'exécution possible, si le service est lancé automatiquement "marche" ou non "arrêt".

Par exemple :

```
# chkconfig --list network
network          0:arrêt 1:arrêt 2:marche          3:marche          4:marche
5:marche         6:arrêt
```

- La commande `service NOM_SERVICE status` permet d'obtenir des informations sur le service pour le niveau d'exécution en cours. Généralement, il est indiqué si le service est démarré ("NOM_SERVICE en cours d'exécution") ou pas ("NOM_SERVICE est arrêté"). D'autres informations peuvent être affichées, par exemple la liste des règles en cours pour le service iptables, ou les interfaces activées pour le service network.

Par exemple :

```
# service network status
Périphériques configurés :
lo em1 p2p1 p3p1
Périphériques actuellement actifs :
lo em1
```

11.2 États d'un service avec systemd

- La commande `systemctl status NOM.service` permet d'afficher plusieurs lignes d'informations sur un service. Ces informations concernent l'état courant du service. Nous nous intéresserons aux lignes préfixées "Loaded" et "Active".

Quelques exemples :

```
# systemctl status iptables.service
iptables.service - IPv4 firewall with iptables
   Loaded: loaded (/lib/systemd/system/iptables.service; enabled)
   Active: active (exited) since Sat, 10 Mar 2012 20:38:28 +0100; 1h 5min
ago
   Process: 1060 ExecStart=/usr/libexec/iptables.init start (code=exited,
status=0/SUCCESS)
   CGroup: name=systemd:/system/iptables.service

# systemctl status NetworkManager.service
NetworkManager.service - Network Manager
   Loaded: loaded (/lib/systemd/system/NetworkManager.service; disabled)
   Active: inactive (dead)
   CGroup: name=systemd:/system/NetworkManager.service

network.service - LSB: Bring up/down networking
   Loaded: loaded (/etc/rc.d/init.d/network)
   Active: active (running) since Sat, 10 Mar 2012 20:38:32 +0100; 1h 8min
ago
   Process: 1170 ExecStart=/etc/rc.d/init.d/network start (code=exited,
status=0/SUCCESS)
   CGroup: name=systemd:/system/network.service
           └─ 1689 /sbin/dhclient -H hostname -l -q -lf
/var/lib/dhclient/dhclient-em1.leases -pf /var/run/dhclient-em1.pid em1
```

- La ligne préfixée "Loaded" permet d'indiquer l'état de chargement du service:

Si le service est configuré, avec le mot clé "loaded", suivi du chemin du fichier de configuration.

Si le service est activé, avec le mot clé "enabled" en fin de ligne.

Si le service est désactivé, avec le mot clé "disabled" en fin de ligne.

Si le service n'est pas configuré ou s'il est en erreur, le mot clé "error" est présent, suivi de la raison.

- Le ligne préfixée "Active" permet d'indiquer l'état d'exécution du service :

Si le service a été lancé avec la mot clé "active".

Si le service n'a pas été lancé avec le mot clé "inactive".

Une deuxième indication, donnée entre parenthèse, permet de préciser l'état du processus en cours pour le service. Avec "running" si ce processus est en cours d'exécution, "exited" si ce processus est terminé, et "dead" si ce processus est en erreur ou s'il n'a pas été lancé.