

GNU/Linux

systemd - Administration du système et des services

Nicolas Delanoue

Université d'Angers - Polytech Angers



POLYTECH[°]
ANGERS



- 1 Usage basique
- 2 Configuration et lancement des services
- 3 Courte introduction aux scripts bash

Commande de supervision du système

Les quelques commandes suivantes permettent de connaître l'état du système :

- `ps` : liste les processus actifs,
- `top` : liste les processus actifs et indique les ressources exploitées (cpu, mem, dots),
- `htop` : comme top mais plus riche,
- `free` : donne la quantité de mémoire libre,
- `df -h` : indique la place disponible sur les disques durs (voir aussi `du`),
- `netstat` : indique les ports utilisés (adresses TCP).

Remarque

Toutes ces commandes tirent les informations des fichiers de `/proc`.

Définition - Service

Un *démon*, aussi appelé *service*, est un processus qui s'exécute en arrière plan.

Exemple de service

apache : serveur http avec lequel des navigateurs (client) http discuteront,

vsftp : serveur ftp avec lequel des clients ftp comme filezilla échangeront des fichiers,

cups : serveur d'impression, vous pouvez avoir plusieurs imprimantes,

lightdm : environnement graphique, si vous l'arrêtez, vous n'aurez qu'un écran en noir et blanc, ...

... : tout ce qui tourne en fond sans faire d'icone dans la barre des tâches...

Gestion des démons

Systemd est un système d'initialisation et un démon qui a été spécifiquement conçu pour le noyau Linux. Il permet, en particulier, de configurer les services qui sont lancés au démarrage.

Encore d'autres démons classiques

`sshd` serveur de connexion à distance,

`proftpd` serveur ftp,

`sendmail` démon SMTP (pour l'envoi d'e-mails),

`cron` outil de lancements périodiques de tâches,

`mpd` music player daemon,

`systemd` le démon qui gère les autres démons...

...

L'outil de management des services

`systemctl` est l'outil principal utilisé pour inspecter et contrôler l'état du système "systemd" et du gestionnaire de service.

Remarque

Vous pouvez par exemple utiliser `systemctl` pour activer/désactiver les services de manière permanente ou uniquement pour la session en cours.

Lister les services actifs

```
user@pc:~$ systemctl list-units --type=service
[...]
mailgraph.service      loaded active running SYSV: mailgraph postfix ...
mariadb.service        loaded active running MariaDB database server
memcached.service      loaded active running Memcached
named.service          loaded active running Berkeley Internet Name ...
network.service        loaded active exited LSB: Bring up/down ...
ssh.service            loaded active running OpenBSD Secure Shell server
[...]
```

ou bien

```
user@pc:~$ systemctl status
```

Connaitre l'état d'un service

```
user@pc:~$ systemctl is-active postfix
active
```


Stop et Start d'un service

```
user@pc:~$ systemctl is-active postfix
active
user@pc:~$ systemctl stop postfix
user@pc:~$ systemctl is-active postfix
inactive
user@pc:~$ systemctl start postfix
user@pc:~$ systemctl is-active postfix
active
```

Activer un service au démarrage

```
$ systemctl enable postfix  
Created symlink from /etc/systemd/system/multi-...
```

Désactiver un service au démarrage

```
$ systemctl disable postfix  
Removed symlink /etc/systemd/system/multi-user...
```

Configuration

Les services se paramètrent avec l'édition de fichiers dans le répertoire `/etc/`. Ce sont souvent des fichiers texte.

Prise en compte des modifications de configuration

restart Relance le service

```
$ systemctl restart postfix
```

reload Recharge le service. En effet, afin de ne pas tuer un service en cours d'utilisation, on préférera taper

```
$ systemctl reload postfix
```

Remarques concernant la configuration des services

- Chaque démon a ses propres fichiers configurations dans un sous dossier de `/etc/` qui porte souvent le nom du démon.
- La syntaxe de ces fichiers change d'un démon à un autre.
- Le plus souvent, on édite partiellement les fichiers par défaut.

Définition

Un *script shell* permet d'automatiser une série d'opérations. Il se présente sous la forme d'un fichier contenant une ou plusieurs commandes qui seront exécutées de manière séquentielle.

Exemple

```
#!/bin/bash
echo "Entrez votre nom svp"
read name
# Affiche le nom passé
echo The name given was :$name
```

Quelques remarques

- La première ligne indique à votre shell qu'elle interpréteur doit prendre en charge le reste du fichier, ici c'est `/bin/bash`
- On n'oubliera pas de rendre le fichier contenant ce script exécutable avec un `chmod +x`.

Paramètres d'un script

- \$0 Nom du script
- \$1 Premier paramètre
- \$2, \$3, etc. Second, troisième paramètre, etc.
- \$* Tous les paramètres
- \$# Nombre d'arguments

Exemple de script nommé script.sh

```
#!/bin/bash
```

```
echo Le premier parametre est : $1
```

```
user@pc: $ ./script.sh toto
```

```
Le premier parametre est : toto
```

Possibilité du langage Bash

- Variable,
- Opération arithmétique,
- Boucle,
- Condition,
- Tableau,
- ...

Url

`https://devhints.io/bash`