



# Démarrage du système sous CentOS 7

24/06/2017

# 1. Le processus de démarrage

Il est important de comprendre le processus de démarrage de Linux pour pouvoir résoudre les problèmes qui peuvent y survenir.

Le processus de démarrage comprend :

## 1.1. Le démarrage du BIOS

Le **BIOS** (Basic Input/Output System) effectue le test **POST** (power on self test) pour détecter, tester et initialiser les composants matériels du système.

Il charge ensuite le **MBR** (Master Boot Record).

## 1.2. Le Master boot record (MBR)

Le Master Boot Record correspond aux 512 premiers bytes du disque de démarrage. Le MBR découvre le périphérique de démarrage et charge le chargeur de démarrage **GRUB2** en mémoire et lui transfère le contrôle.

Les 64 bytes suivants contiennent la table de partition du disque.

## 1.3. Le chargeur de démarrage GRUB2 (Bootloader)

Le chargeur de démarrage par défaut de la distribution CentOS 7 est **GRUB2** (GRand Unified Bootloader). GRUB2 remplace l'ancien chargeur de démarrage Grub (appelé également GRUB legacy).

Le fichier de configuration de GRUB 2 se situe sous **/boot/grub2/grub.cfg** mais ce fichier ne doit pas être directement édité.

Les paramètres de configuration du menu de GRUB2 se trouvent sous **/etc/default/grub** et servent à la génération du fichier grub.cfg.

*Exemple de fichier /etc/default/grub*

```
# cat /etc/default/grub
GRUB_TIMEOUT=5
GRUB_DEFAULT=saved
GRUB_DISABLE_SUBMENU=true
GRUB_TERMINAL_OUTPUT="console"
GRUB_CMDLINE_LINUX="rd.lvm.lv=rhel/swap crashkernel=auto rd.lvm.lv=rhel/root rhgb
quiet net.ifnames=0"
GRUB_DISABLE_RECOVERY="true"
```

Si des changements sont effectués à un ou plusieurs de ces paramètres, il faut lancer la commande

**grub2-mkconfig** pour régénérer le fichier `/boot/grub2/grub.cfg`.

```
[root] # grub2-mkconfig -o /boot/grub2/grub.cfg
```

- GRUB2 cherche l'image du noyau compressé (le fichier `vmlinuz`) dans le répertoire `/boot`.
- GRUB2 charge l'image du noyau en mémoire et extrait le contenu du fichier image `initramfs` dans un dossier temporaire en mémoire en utilisant le système de fichier `tmpfs`.

## 1.4. Le noyau

Le noyau démarre le processus **systemd** avec le PID 1.

```
root          1          0  0 02:10 ?          00:00:02 /usr/lib/systemd/systemd --switched
-root --system --deserialize 23
```

## 1.5. systemd

Systemd est le père de tous les processus du système. Il lit la cible du lien `/etc/systemd/system/default.target` (par exemple `/usr/lib/systemd/system/multi-user.target`) pour déterminer la cible par défaut du système. Le fichier définit les services à démarrer.

Systemd positionne ensuite le système dans l'état défini par la cible en effectuant les tâches d'initialisations suivantes :

1. Paramétrer le nom de machine
2. Initialiser le réseau
3. Initialiser SELinux
4. Afficher la bannière de bienvenue
5. Initialiser le matériel en se basant sur les arguments fournis au kernel lors du démarrage
6. Monter les systèmes de fichiers, en incluant les systèmes de fichiers virtuels comme `/proc`
7. Nettoyer les répertoires dans `/var`
8. Démarrer la mémoire virtuelle (swap)

# 2. Protéger le chargeur de démarrage GRUB2

Pourquoi protéger le chargeur de démarrage avec un mot de passe ?

1. Prévenir les accès au mode utilisateur **Single** – Si un attaquant peut démarrer en mode single user, il devient l'utilisateur root.
2. Prévenir les accès à la console GRUB – Si un attaquant parvient à utiliser la console GRUB, il

peut changer sa configuration ou collecter des informations sur le système en utilisant la commande `cat`.

3. Prévenir les accès à des systèmes d'exploitation non sécurisés. S'il y a un double boot sur le système, un attaquant peut sélectionner au démarrage un système d'exploitation comme DOS qui ignore les contrôles d'accès et les permissions des fichiers.

Pour protéger par mot de passe le GRUB2 :

- Retirer **`-unrestricted`** depuis la déclaration principale **`CLASS=`** dans le fichier **`/etc/grub.d/10_linux`**.
- Si un utilisateur n'a pas encore été configuré, utiliser la commande **`grub2-setpassword`** pour fournir un mot de passe à l'utilisateur root :

```
# grub2-setpassword
```

Un fichier **`/boot/grub2/user.cfg`** va être créé s'il n'était pas encore présent. Il contient le mot de passe hashé du GRUB.



Cet commande ne supporte que les configurations avec un seul utilisateur root.

*Exemple de fichier `/boot/grub2/user.cfg`*

```
[root]# cat /boot/grub2/user.cfg
GRUB2_PASSWORD=grub.pbkdf2.sha512.10000.CC6F56....A21
```

- Recréer le fichier de configuration avec la commande **`grub2-mkconfig`** :

```
[root]# grub2-mkconfig -o /boot/grub2/grub.cfg
Generating grub configuration file ...
Found linux image: /boot/vmlinuz-3.10.0-327.el7.x86_64
Found initrd image: /boot/initramfs-3.10.0-327.el7.x86_64.img
Found linux image: /boot/vmlinuz-0-rescue-f9725b0c842348ce9e0bc81968cf7181
Found initrd image: /boot/initramfs-0-rescue-f9725b0c842348ce9e0bc81968cf7181.img
done
```

- Redémarrer le serveur et vérifier.

Toutes les entrées définies dans le menu du GRUB vont maintenant nécessiter la saisie d'un utilisateur et d'un mot de passe à chaque démarrage. Le système ne démarrera pas de noyau sans intervention directe de l'utilisateur depuis la console.

- Lorsque l'utilisateur est demandé, saisir "root" ;
- Lorsqu'un mot de passe est demandé, saisir le mot de passe fourni à la commande `grub2-setpassword`.

Pour ne protéger que l'édition des entrées du menu de GRUB et l'accès à la console, l'exécution de la commande `grub2-setpassword` est suffisante.

### 3. Systemd

**Systemd** est un gestionnaire de service pour les systèmes d'exploitation Linux.

Il est développé pour :

- rester compatible avec les anciens scripts d'initialisation SysV,
- fournir de nombreuses fonctionnalités comme le démarrage en parallèle des services systèmes au démarrage du système, l'activation à la demande de démons, le support des instantanés ou la gestion des dépendances entre les services.



Systemd est le système d'initialisation par défaut depuis la RedHat/CentOS 7.

Systemd introduit le concept d'unités systemd.

*Table 1. Principaux types d'unités systemd disponibles*

| Type             | Extension du fichier | Observation  |
|------------------|----------------------|--|
| Unité de service | .service             | Service système  |
| Unité cible      | .target              | Un groupe d'unités systemd                               |
| Unité automount  | .automount           | Un point de montage automatique pour système de fichiers |



Il existe de nombreux types d'unités : Device unit, Mount unit, Path unit, Scope unit, Slice unit, Snapshot unit, Socket unit, Swap unit, Timer unit.

- Systemd supporte les instantanés de l'état du système et leur restauration.
- Les points de montage peuvent être configurés comme des cibles systemd.
- Au démarrage, systemd crée des sockets en écoute pour tous les services systèmes qui supportent ce type d'activation et passe ces sockets à ces services aussitôt qu'elles sont démarrées. Cela rend possible la relance d'un service sans perdre un seul message qui lui est envoyé par le réseau durant son indisponibilité. La socket correspondante reste accessible et tous les messages sont mis en file d'attente.
- Les services systèmes qui utilisent D-BUS pour leurs communications inter-process peuvent être démarrés à la demande dès la première utilisation par un client.
- Systemd stoppe ou relance uniquement les services en cours de fonctionnement. Les versions précédentes de CentOS tentaient directement de stopper les services sans vérifier leur état en cours.
- Les services systèmes n'héritent d'aucun contexte (comme les variables d'environnements

HOME et PATH). Chaque service fonctionne dans son propre contexte d'exécution.

Toutes les opérations des unités de service sont soumises à un timeout par défaut de 5 minutes pour empêcher un service malfunctionnant de geler le système.

## 3.1. Gérer les services systèmes

Les unités de service se terminent par l'extension de fichier .service et ont un but similaire à celui des scripts init. La commande systemctl est utilisée pour afficher, lancer, arrêter, redémarrer, activer ou désactiver des services système.



Les commandes service et chkconfig sont toujours disponibles dans le système et fonctionnent comme prévu, mais sont uniquement incluses pour des raisons de compatibilité et doivent être évitées.

Table 2. Comparaison des utilitaires service et systemctl

| service                         | systemctl                                 | Description                                  |
|---------------------------------|---|--|
| service <i>name</i> start       | systemctl start <i>name.service</i>       | Lancer un service                            |
| service <i>name</i> stop        | systemctl stop <i>name.service</i>        | Stoppe un service                            |
| service <i>name</i> restart     | systemctl restart <i>name.service</i>     | Relance un service                           |
| service <i>name</i> reload      | systemctl reload <i>name.service</i>      | Recharge une configuration                   |
| service <i>name</i> status      | systemctl status <i>name.service</i>      | Vérifie si un service fonctionne             |
| service <i>name</i> condrestart | systemctl try-restart <i>name.service</i> | Relance un service seulement s'il fonctionne |
| service --status-all            | systemctl list-units --type service --all | Affiche le status de tous les services       |

Table 3. Comparaison des utilitaires chkconfig et systemctl

| chkconfig                    | systemctl                                | Description  |
|------------------------------|--|--|
| chkconfig <i>name</i> on     | systemctl enable <i>name.service</i>     | Active un service  |
| chkconfig <i>name</i> off    | systemctl disable <i>name.service</i>    | Désactive un service                                     |
| chkconfig --list <i>name</i> | systemctl status <i>name.service</i>     | Vérifie si un service fonctionne                         |
| chkconfig --list             | systemctl list-unit-files --type service | Liste tous les services et vérifie s'ils fonctionnent    |
| chkconfig --list             | systemctl list-dependencies --after      | Liste les services qui démarrent avant l'unité spécifiée |
| chkconfig --list             | systemctl list-dependencies --before     | Liste les services qui démarrent après l'unité spécifiée |

Exemples :

```
systemctl stop nfs-server.service  
# ou  
systemctl stop nfs-server
```

Pour lister toutes les unités chargées actuellement :

```
systemctl list-units --type service
```

Pour lister toutes les unités pour vérifier si elles sont activées :

```
systemctl list-unit-files --type service
```

```
systemctl enable httpd.service  
systemctl disable bluetooth.service
```

## 3.2. Exemple de fichier .service pour le service postfix

postfix.service Unit File  
What follows is the content of the /usr/lib/systemd/system/postfix.service unit file as currently provided by the postfix package:

```
[Unit]  
Description=Postfix Mail Transport Agent  
After=syslog.target network.target  
Conflicts=sendmail.service exim.service  
  
[Service]  
Type=forking  
PIDFile=/var/spool/postfix/pid/master.pid  
EnvironmentFile=-/etc/sysconfig/network  
ExecStartPre=-/usr/libexec/postfix/aliasesdb  
ExecStartPre=-/usr/libexec/postfix/chroot-update  
ExecStart=/usr/sbin/postfix start  
ExecReload=/usr/sbin/postfix reload  
ExecStop=/usr/sbin/postfix stop  
  
[Install]  
WantedBy=multi-user.target
```

### 3.3. Utiliser les cibles systèmes

Sur CentOS7/RHEL7, le concept des niveaux d'exécution a été remplacé par les cibles Systemd.

Les cibles Systemd sont représentées par des unités de cible (target units). Les unités de cible se terminent par l'extension de fichier .target et leur unique but consiste à regrouper d'autres unités Systemd dans une chaîne de dépendances.

Par exemple, l'unité **graphical.target**, qui est utilisée pour lancer une session graphique, lance des services systèmes comme le gestionnaire d'affichage GNOME (gdm.service) ou le services des comptes (accounts-daemon.service) et active également l'unité multi-user.target.

De manière similaire, l'unité multi-user.target lance d'autres services système essentiels, tels que NetworkManager (NetworkManager.service) ou D-Bus (dbus.service) et active une autre unité cible nommée basic.target.

Table 4. Comparaison des targets systemctl et runlevel

| Runlevel | Target Units      | Description  |
|----------|-------------------|--|
| 0        | poweroff.target   | Arrête le système et l'éteint                                |
| 1        | rescue.target     | Active un shell de secours                                   |
| 2        | multi-user.target | Active un système multi-utilisateur sans interface graphique |
| 3        | multi-user.target | Active un système multi-utilisateur sans interface graphique |
| 4        | multi-user.target | Active un système multi-utilisateur sans interface graphique |
| 5        | graphical.target  | Active un système multi-utilisateur avec interface graphique |
| 6        | reboot.target     | Arrête puis redémarre le système                             |

#### 3.3.1. La cible par défaut

Pour déterminer quelle cible est utilisée par défaut :

```
systemctl get-default
```

Cette commande recherche la cible du lien symbolique située à /etc/systemd/system/default.target et affiche le résultat.



```
$ systemctl get-default
graphical.target
```

La commande systemctl peut également fournir la liste des cibles disponibles :

*Lister toutes les cibles disponibles*

```
sudo systemctl list-units --type target
UNIT                                LOAD    ACTIVE SUB    DESCRIPTION
basic.target                       loaded active active Basic System
bluetooth.target                   loaded active active Bluetooth
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
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-user-lookup.target             loaded active active User and Group Name Lookups
paths.target                       loaded active active Paths
remote-fs.target                   loaded active active Remote File Systems
slices.target                      loaded active active Slices
sockets.target                     loaded active active Sockets
sound.target                       loaded active active Sound Card
swap.target                        loaded active active Swap
sysinit.target                     loaded active active System Initialization
timers.target                      loaded active active Timers
```

Pour configurer le système afin d'utiliser une cible différente par défaut :

```
systemctl set-default name.target
```

Exemple :

```
[root]# systemctl set-default multi-user.target
rm '/etc/systemd/system/default.target'
ln -s '/usr/lib/systemd/system/multi-user.target' '/etc/systemd/system/default.target'
```

Pour passer à une unité de cible différente dans la session actuelle :

```
systemctl isolate name.target
```

Le **mode de secours** ("Rescue mode") fournit un environnement simple et permet de réparer votre système dans les cas où il est impossible d'effectuer un processus de démarrage normal.

En mode de secours, le système tente de monter tous les systèmes de fichiers locaux et de lancer plusieurs services système importants, mais n'active pas d'interface réseau ou ne permet pas à d'autres utilisateurs de se connecter au système au même moment.

Sur CentOS7/RHEL7, le mode de secours est équivalent au mode utilisateur seul (single user mode) et requiert le mot de passe root.

Pour modifier la cible actuelle et entrer en mode de secours dans la session actuelle :

```
systemctl rescue
```

Le **mode d'urgence** ("Emergency mode") fournit l'environnement le plus minimaliste possible et permet de réparer le système même dans des situations où le système est incapable d'entrer en mode de secours. Dans le mode d'urgence, le système monte le système de fichiers root uniquement en lecture, il ne tentera pas de monter d'autre système de fichiers locaux, n'activera pas d'interface réseau et lancera quelques services essentiels.

Pour modifier la cible actuelle et entrer en mode d'urgence dans la session actuelle :

```
systemctl emergency
```

### 3.3.2. Arrêt, suspension et hibernation

La commande `systemctl` remplace un certain nombre de commandes de gestion de l'alimentation utilisées dans des versions précédentes :

Table 5. Comparaison entre les commandes de gestion de l'alimentation et `systemctl`

| Ancienne commande | Nouvelle commande      | Description                               |
|-------------------|------------------------|---|
| halt              | systemctl halt         | Arrête le système.                        |
| poweroff          | systemctl poweroff     | Met le système hors-tension.              |
| reboot            | systemctl reboot       | Redémarre le système.                     |
| pm-suspend        | systemctl suspend      | Suspend le système.                       |
| pm-hibernate      | systemctl hibernate    | Met le système en hibernation.            |
| pm-suspend-hybrid | systemctl hybrid-sleep | Met en hibernation et suspend le système. |

## 3.4. Le processus journald

Les fichiers journaux peuvent, en plus de `rsyslogd`, également être gérés par le démon **journald** qui est un composant de `systemd`.

Le démon `journald` capture les messages Syslog, les messages du journal du noyau, les messages du disque RAM initial et du début du démarrage, ainsi que les messages inscrits sur la sortie standard et la sortie d'erreur standard de tous les services, puis il les indexe et les rend disponibles à l'utilisateur.

Le format du fichier journal natif, qui est un fichier binaire structuré et indexé, améliore les recherches et permet une opération plus rapide, celui-ci stocke également des informations de métadonnées, comme l'horodatage ou les ID d'utilisateurs.

## 3.5. La commande `journalctl`

La commande **`journalctl`** permet d'afficher les fichiers journaux.

```
journalctl
```

La commande liste tous les fichiers journaux générés sur le système. La structure de cette sortie est similaire à celle utilisée dans `/var/log/messages/` mais elle offre quelques améliorations :

- la priorité des entrées est marquée visuellement ;
- les horodatages sont convertis au fuseau horaire local de votre système ;
- toutes les données journalisées sont affichées, y compris les journaux rotatifs ;
- le début d'un démarrage est marqué d'une ligne spéciale.

### 3.5.1. Utiliser l'affichage continu

Avec l'affichage continu, les messages journaux sont affichés en temps réel.

```
journalctl -f
```

Cette commande retourne une liste des dix lignes de journal les plus récentes. L'utilitaire `journalctl` continue ensuite de s'exécuter et attend que de nouveaux changements se produisent pour les afficher immédiatement.

### 3.5.2. Filtrer les messages

Il est possible d'utiliser différentes méthodes de filtrage pour extraire des informations qui correspondent aux différents besoins. Les messages journaux sont souvent utilisés pour suivre des comportements erronés sur le système. Pour afficher les entrées avec une priorité sélectionnée ou plus élevée :

```
journalctl -p priority
```

Il faut remplacer `priority` par l'un des mots-clés suivants (ou par un chiffre) :

- `debug (7)`,
- `info (6)`,
- `notice (5)`,
- `warning (4)`,
- `err (3)`,
- `crit (2)`,
- `alert (1)`,
- et `emerg (0)`.