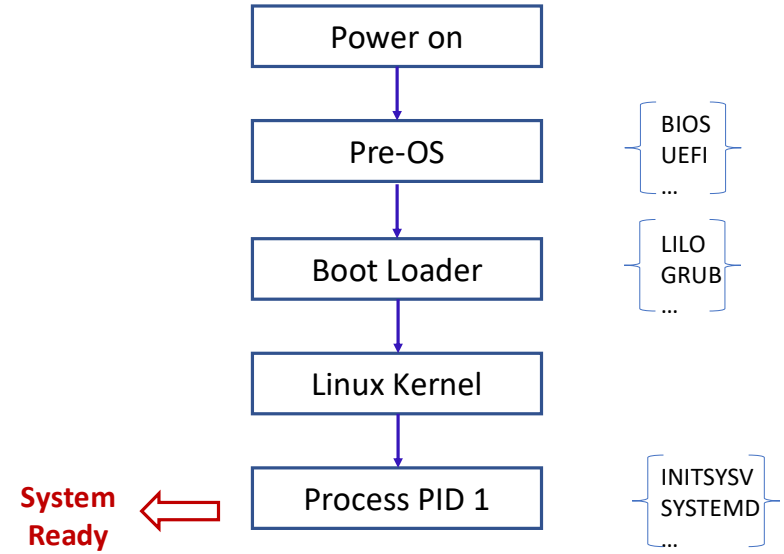
Démarrage et arrêt du système Linux

- Introduction
- Processus de démarrage
- Démarrage: BIOS
- Chargeur de démarrage: BootLoader
- Démarrage du noyau
- Lancement du processus PID 1
- Gestionnaires de services
- Arrêt du système

Introduction

- Le système informatique subit plusieurs phases de **processus d'amorçage** (« boot strap process ») depuis l'événement de mise sous tension jusqu'à ce qu'il offre à l'utilisateur un système d'exploitation (OS) **pleinement fonctionnel**.
  - ⇒ Pour qu'un système GNU/Linux soit utilisable, il doit d'abord passer par **plusieurs étapes d'initialisation** et de lancement de divers programmes.
  - ⇒ Lorsque l'ordinateur démarre, les messages défilant sur la console révèlent de nombreuses **initialisations et configurations** automatiques. Parfois, il est souhaitable de **modifier légèrement le déroulement de cette étape**, ce qui implique de bien la comprendre.

Processus de démarrage

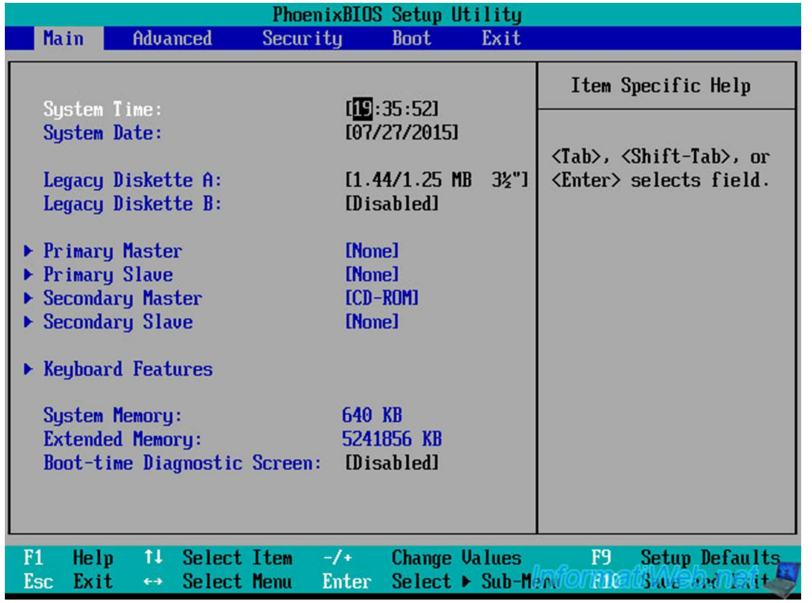


Démarrage: BIOS (1)

- **BIOS: B**asic **I**nput **O**utput **S**ystem
- Se situe sur **la ROM** de la carte mère
- Le BIOS fait quelques tests (**POST : Pre-Operating System Tests**) sur le matériel. A ce stade, on peut arrêter le BIOS pour le configurer (selon les bios, ex : touche DEL).
- **Analyse la configuration matérielle** de l'ordinateur: Recensement des périphériques + Test de certains périphériques :
  - ⇒ Test de la mémoire,
  - ⇒ Test de la présence de clavier, ...
  - ⇒ Test de la présence des disques durs, lecteurs de CDROM …
- **Localisation de l'OS:** il cherche le système d'exploitation sur le premier périphérique disponible.

Démarrage: BIOS (2)

Démarrage: BIOS (3)



Démarrage: BIOS (4)

BIOS vs UEFI

- **UEFI: U**nified **E**xtensible **F**irmware **I**nterface, la dernière méthode de démarrage d'un ordinateur conçue pour **remplacer le BIOS.**
- **UEFI** est généralement utilisé sur les systèmes 64 bits ultérieurs à Windows 7
- la plupart des nouvelles cartes mères permettent également aux utilisateurs de passer en mode de compatibilité **Legacy + UEFI**.
- Lorsque le **BIOS** détecte un système installé sous **Legacy**, il lance le mode de démarrage Legacy. De même, s'il détecte un système installé sous **UEFI,** il démarrera en mode **UEFI**.

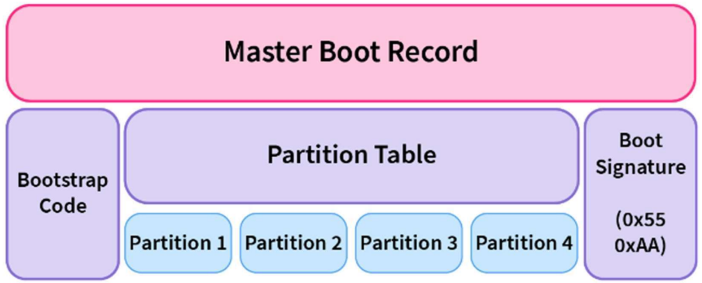
Démarrage: BIOS (5)

BIOS vs UEFI

- Le **UEFI** est un programme qui est conçu pour **remplacer le BIOS**
- Il tient plus du système d'exploitation que du BIOS:
  - ⇒ Interface graphique à fenêtre
  - ⇒ Accès à Internet
  - ⇒ Mesures de sécurité et anti-virus intégrés
  - ⇒ Support **GPT** (GUID Partition Table—remplacement du MBR) pour démarrer sur des partitions de plus de 2TB
  - ⇒ Architecture modulaire et une grande partie est **écrite en C** au lieu d'en assembleur ce qui rend l'adaptation pour d'autres plateformes plus facile

Démarrage: BIOS (5)

- Lancement de l'OS:
  ⇒ lit les 512 premiers octets du disque dur:
  Ces 512 octets constituent le **MBR = Master Boot Record**
- Ce secteur contient:
  ⇒ **un boot Loader**
  ⇒ **la table des partitions**

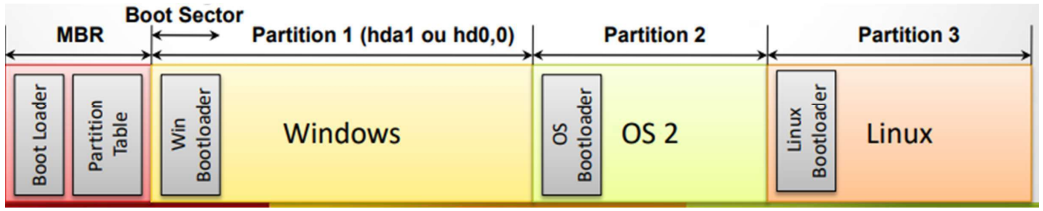


Chargeur de démarrage (Boot Loader) (1)

- Pour qu'un système Unix puisse se lancer, il faut que **le noyau soit chargé en mémoire et qu'il s'exécute**
- Le but principal d'un **boot loader** est de charger le noyau en mémoire et de le lancer
- Un bootloader peut permettre, aussi, de
  ⇒ Choisir entre plusieurs noyaux à charger
  ⇒ Passer des paramètres au noyau chargé
  ⇒ Choisir entre plusieurs OS (sur un système multi OS)
- Le boot loader **est obligatoire** pour charger le noyau
- Le boot loader fait la transition entre le démarrage matériel de la machine (**mise sous tension**) et l'exécution du noyau (**lancement de l'OS**)

Chargeur de démarrage (Boot Loader) (2)

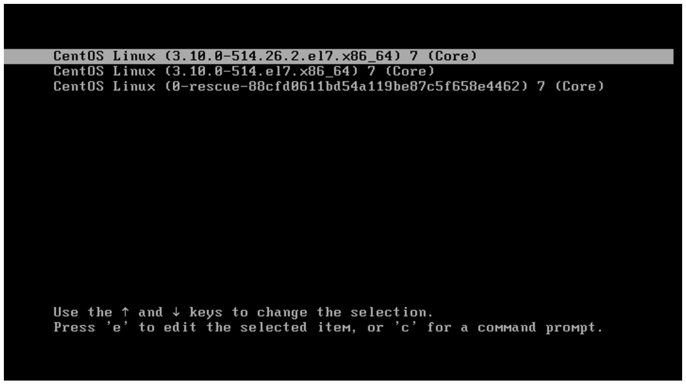
1. **Le BIOS** du système examine le système et lance **le chargeur de démarrage** de **l'Étape 1 (code d'amorçage)** sur le bloc de démarrage maître (MBR) du disque dur principal.
2. **Le chargeur de démarrage** de **l'Étape 1** se charge en mémoire et lance **le chargeur de démarrage** de **l'Étape 2** à partir de la partition /boot/.
3. **Le chargeur de démarrage** de **l'Étape 2** charge en **mémoire le noyau** qui à son tour charge tous les modules nécessaires et monte la partition root en lecture-seule.



Chargeur de démarrage (Boot Loader) (3)

- Le chargeur de démarrage **localise le noyau** du système d'exploitation sur le disque, le charge et l'exécute
- Lorsque l' ordinateur héberge **plusieurs systèmes** (multi-amorçage), il permet à l'utilisateur de **choisir quel système démarrer.**

- Types de chargeurs:
  ⇒ **Grub** : GRand Unified Bootloader (Première version)
  ⇒ **Grub2** : nouvelle version
  ⇒ **Lilo** : LInux LOader (délaissé par les développeurs) ;



Chargeur de démarrage (Boot Loader) (4)

Mise en place de GRUB 2

- La plupart des distributions récentes **installent automatiquement** GRUB2.
- Cette étape n'est alors nécessaire que pour réinstaller GRUB2 ou migrer de **Grub Legacy** vers **Grub2** dans le cas d'une ancienne distribution.
- La procédure de mise en place comprend :
  - ⇒ L'installation ou mise à jour des paquetages
  - ⇒ L'installation du chargeur
  - ⇒ Configuration

Chargeur de démarrage (Boot Loader)

Mise en place de GRUB 2

- L'installation des paquetages peut se faire à l'aide de la commande :

```
yum -y install grub2 os-prober
```

ou

```
yum -y update grub2 os-prober
```

pour la mise à jour

- Le paquetage **os-prober** est utilisé particulièrement par **grub2** pour chercher les systèmes installés et générer les entrées correspondantes dans le fichier **grub.cfg**

Chargeur de démarrage (Boot Loader) (5)

- **Fichiers Grub2:**

⇒ **/etc/default/grub** - le fichier contenant les **paramètres du menu** de GRUB 2,

⇒ **/etc/grub.d/** - le répertoire contenant **les scripts de création du menu GRUB 2**, permettant notamment de personnaliser le menu de démarrage,

⇒ **/boot/grub2/grub.cfg** - le **fichier de configuration final** de GRUB 2, **non modifiable**. **(/boot/grub/grub.cfg** sous Debian).

⇒ Le dernier fichier est généré automatiquement par le programme **grub2-mkconfig** à partir des scripts **/etc/default/grub** et **/etc/grub.d/**

Chargeur de démarrage (Boot Loader) (6)

Configuration de GRUB 2

- Avec **grub2** on n'édite pas directement le fichier **grub.cfg**, mais on intervient sur :
  - ⇒ **/etc/default/grub** : qui contient les valeurs de quelques paramètres, comme la résolution de l'écran, le Timeout, le système par défaut, ...
  - ⇒ **/etc/grub.d/*** : un ensemble de scripts permettant de générer le fichier **grub.cfg**. Et dans lesquels on peut rajouter par exemple le code permettant de mettre en place un mot de passe de démarrage.
- Après modification, la commande **grub2-mkconfig** doit être appelée pour régénérer un nouveau **grub.conf**

Chargeur de démarrage (Boot Loader) (7)

Fichier /etc/default/grub

- Contient les valeurs associées aux **paramètres de démarrage**
- Chacune de ses lignes est sous forme : **VARIABLE=valeur**

Exemple:

```
GRUB_TIMEOUT=5
GRUB_DISTRIBUTOR="$(sed 's, release .*$,,g' /etc/system-
release)"
GRUB_DEFAULT=saved
GRUB_DISABLE_SUBMENU=true
GRUB_TERMINAL_OUTPUT="console"
GRUB_CMDLINE_LINUX="crashkernel=auto rd.lvm.lv=centos/root
rd.lvm.lv=centos/swap rhgb quiet"
GRUB_DISABLE_RECOVERY="true"
```


Chargeur de démarrage (Boot Loader) (8)

Fichier /etc/default/grub: quelques paramètres

- **GRUB_TIMEOUT : durée en seconde** à attendre avant le démarrage du système par défaut
- **GRUB_DEFAULT :** désigne **le système à démarrer par défaut**, sa valeur peut être l'ordre du système dans le menu (0 est le premier) son nom ou le mot clé **saved** pour désigner le dernier système activé
- **GRUB_DISTRIBUTOR**: Défini par les distributeurs de GRUB et utilisé pour générer des titres d'entrée de menu plus informatifs. Par exemple :

```
GRUB_DISTRIBUTOR=`echo -n TITRE PERSONNALISÉ`#donne : TITRE
PERSONNALISÉ GNU/Linux
```

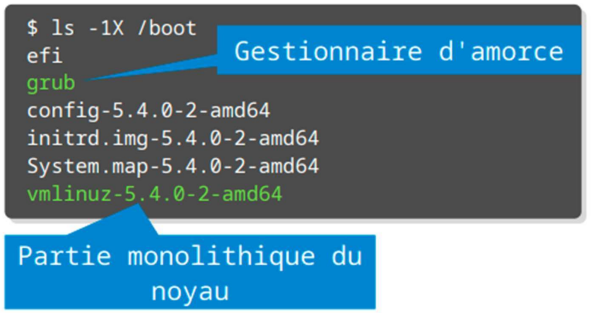

Chargeur de démarrage (Boot Loader) (9)

Fichier /etc/default/grub: quelques paramètres

- **GRUB_DISABLE_SUBMENU** : si on souhaite voir directement une entrée de menu par noyau linux disponible (On peut supprimer les noyaux les plus anciens → Nettoyer le système)
- **GRUB_CMDLINE_LINUX :** les arguments à passer au noyau Linux
- **GRUB_DISABLE_RECOVERY** : la valeurs **true** de ce paramètre permet de désactiver la génération de l'entrée du menu relative au mode **Recovery**
- **GRUB_BACKGROUND:** Paramètre à ajouter si vous voulez mettre une image en arrière plan.

Démarrage du noyau

- **Le boot loader charge et exécute le noyau sélectionné** et l'image **initrd** ou **initramfs** = *INITial RAM filesystem*).
- Le fichier du noyau s'appelle en général **vmlinux** sur linux (vmlinuz est sa version compressée autoextractible).



/boot: fichiers de configuration du démarrage (boot), noyaux et d'autres fichiers indispensables au moment du démarrage (boot).

Démarrage du noyau

- Pour pouvoir démarrer les applications, il faut pouvoir lire les exécutables depuis **un système de fichiers**
- Le noyau doit:
  - ⇒ Localiser le **système de fichiers racine (/)** et le monter .
  - ⇒ Disposer **du/des pilote(s)** de périphérique nécessaire(s) pour accéder au système de fichiers
- Si tous les pilotes nécessaires ont été compilés en dur dans le noyau **== > Noyau très volumineux**
- Le plus souvent, **les pilotes** sont généralement compilés sous forme de modules (afin de pouvoir charger uniquement ceux **utiles** dans la configuration donnée)
- Or les modules sont des fichiers et pour pouvoir les charger, il faut que le système de fichiers sur lequel ils sont **soit lui-même monté...**

Démarrage du noyau

Utilité d'un système de fichiers initial

- Donc, dans de nombreuses distributions classiques, **les pilotes de périphérique nécessaires pour accéder au système de fichiers** racine ne sont pas compilés en dur
- Donc le noyau **ne peut pas monter ce système de fichiers racine...**
- La solution passe par une image de système de fichiers racine initial (**initrd/initramfs**)
- **L'image est chargée en mémoire** par le bootloader **et son adresse est passée au noyau**

Démarrage du noyau

Initrd -- initramfs

- **Initramfs** est utilisé par le noyau comme système de fichiers racine **temporaire** jusqu'à ce que le noyau soit démarré et que le vrai système de fichiers racine soit monté. Il contient les pilotes nécessaires compilé à l'intérieur, ce qui l'aide à accéder aux partitions du disque dur et d'autres matériels
- **Initrd** est un ancien nom: actuellement **initramfs**
- Il contient **le strict minimum** qui peut être requis par le noyau pour charger le « **vrai** » système de fichiers racine :
  - ⇒ il peut s'agir de modules de pilotes pour les disques durs ou d'autres périphériques **sans lesquels le système ne peut pas démarrer**, ou, plus fréquemment, des modules et des scripts d'initialisation permettant d'ouvrir des partitions chiffrées, …

Lancement du Processus PID 1

- A ce stade, le noyau **libère l'espace utilisé précédemment** pour la détection et la configuration des périphériques
- Ensuite, il lance le premier processus **(PID 1)**
- Sur un système UNIX, **le tout premier processus (PID 1)** a un rôle particulier :
  - ⇒ **c'est le seul qui est lancé par le noyau** ; il doit donc assurer toute l'initialisation du système, le lancement des différents services ;
  - ⇒ Le noyau **panique** s'il n'est pas capable de le trouver (**Kernel panic - not syncing: No working init found**.)

Gestionnaires de système et de services

- Le processus **PID 1** est donc le processus d'initialisation du système et de gestion des services
- Différents **systèmes de gestion des services** sont utilisés :
  - ⇒ **SysVinit:** processus init historique, hérité de UNIX System V.
  - ⇒ **Upstart:** sorti en 2006, à l'origine pour Ubuntu.
  - ⇒ **Systemd:** sorti en 2010.
- La commande : **pstree** permet d'afficher l'arbre des processus. Celui au sommet est **le gestionnaire de services utilisé**

Gestionnaire de service SysVinit

- **SysVinit** est le système d'initialisation hérité d'UNIX Système V
- Ce système est représenté par le processus **/sbin/init**
- Le processus **/sbin/init** a le **pid = 1**
- Il est le parent de tous les processus présents sur le système
- Le processus **init** effectue un contrôle des partitions puis procède au montage du système de fichier principal sous **/**
- Après, **init** lance les différents services suivant le contenu du fichier **/etc/inittab** en faisant la distinction entre les différents **niveaux d'exécution (runlevels)**

Gestionnaire de service SysVinit

Niveaux d'exécution (1)

- Le fonctionnement d'un système Linux régi par un ensemble de **niveaux d'exécution**
- La gestion des niveaux d'exécution consiste à déterminer quel doit être **le comportement du système** quand il entre dans un niveau donné.

Les niveaux d'exécution représentent différents modes dans lesquels un ordinateur peut fonctionner. Chaque niveau d'exécution définit un ensemble spécifique de services ou processus qui doivent être exécutés ou arrêtés lors du démarrage ou de l'arrêt du système.

Gestionnaire de service SysVinit

Niveaux d'exécution (2)

- La distribution RedHat distingue les niveaux d'exécution suivants :
  - ⇒ **Le niveau 0: arrêt:** correspond à l'arrêt du système, et aucun service n'est disponible.
  - ⇒ **Le niveau 1,s,S: mode utilisateur unique:** réservé aux opérations de maintenance et ne permet qu'une seule connexion (compte root). La plus part des services sont arrêtés dans ce niveau (le système a une activité minimum). Parfait pour l'administrateur qui souhaite effectuer des opérations de maintenance.
  - ⇒ **Le niveau 2: non utilisé- peut être défini par l'utilisateur**

Niveaux d'exécution (3)

⇒ **Le niveau 3: mode multi-utilisateurs complet:** Tous les services nécessaires sont démarrés et plusieurs utilisateurs peuvent se connecter **en mode texte (**l'interface graphique n'est pas disponible)

⇒ **Le niveau 4: non utilisé- peut être défini par l'utilisateur**

⇒ **Le niveau 5: mode multi-utilisateurs graphique:** Identique au mode 3, mais les utilisateurs peuvent se connecter en mode graphique et disposer d'un gestionnaire de fenêtre.

⇒ **Le niveau 6: redémarrage:** Passer dans ce niveau redémarre la machine. Après le démarrage, le système se trouve dans le mode indiqué par **initdefault** dans **/etc/inittab**.

Niveaux d'exécution: commandes

- La commande : `init N` ou `telinit N` permet de faire passer le système au niveau passé en argument
- La commande `runlevel` ou `who -r` permet d'afficher le niveau courant ainsi que le précédent

Fichier /etc/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 évènements arrivent
- Il y est indiqué quels sont les scripts qui doivent être exécutés lors du **changement de niveau d'exécution**

Fichier /etc/inittab: structure

- Typiquement, une entrée dans le fichier **/etc/inittab** a la forme suivante :

```
id:niveaux_exécution:action:processus
```

  - ⇒ **Id:** Séquence unique de 1 à 4 caractères qui identifient une entrée dans inittab
  - ⇒ **niveaux_exécution:** Liste des niveaux d'exécution pour lesquels l'action doit être faite.
  - ⇒ **action:** Description de l'action à faire.
  - ⇒ **processus:** Commande à exécuter.

Gestionnaire de service SysVinit

Fichier /etc/inittab: Exemple

```
# The default runlevel.
:5:initdefault:

# Boot-time system
configuration/initialization
script.

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

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

- **Le niveau 5** est le niveau par défaut
- La combinaison **Controle-alt-del** lancera un "shutdown" immédiat puis de /etc/init.d/rc 2

Gestionnaire de service SysVinit

Répertoires /etc/rcX.d

- Le fichier **inittab** précédent nous indique qu'à l'entrée dans un niveau d'exécution donné **(ex. 3),** le script **/etc/rc.d/rc** est exécuté avec l'argument **3**
- Ce script va aller dans le répertoire **/etc/rc.d/rcx.d** (où x est remplacé par l'argument, ici 3)
- Dans ce répertoire, on trouve des fichiers (le plus souvent les liens symboliques) dont le nom est de la forme **Knnwwwwww** (où nn est un nombre et **wwwwww** un nom quelconque) ou **Snnwwwwww**
- Le plus souvent, ces fichiers sont des liens symboliques vers des scripts contenus dans **/etc/rc.d/init.d**
- Ces scripts portent **le nom du service qu'ils gèrent** et vont réagir à l'argument **start** ou **stop** en **démarrant/arrêtant le service concerné**

Gestionnaire de service SysVinit: Résumé

- Le système init:
  - ⇒ **Un niveau d'exécution** (runlevel) défini par **/etc/inittab**
  - ⇒ **Des scripts** dans **/etc/init.d** lançant les processus
  - ⇒ **Des répertoires /etc/rcx.d**, pour x = 0,1,...,6 contenant des liens symboliques vers les scripts de /etc/init.d

init est remplacé récemment: système vieillissant !!

Gestionnaire de service SysVinit: inconvénients

- Malheureusement, **SysVinit** reste très basique, peu performant et sujet à de nombreux problèmes. Il n'est pas très adapté à toutes les évolutions modernes que connaît Linux.
- Ces principaux défauts :
  - ⇒ Le démarrage des services est **séquentiel**.
  - ⇒ La prise en charge des **dépendances entre services** est très basique
  - ⇒ Tout ou presque est sous forme de scripts Shell ; ceci rend le processus de démarrage **lent** et **fastidieux**.
  - ⇒ ...
- Plusieurs alternatives ont été développées et **systemd** s'est progressivement imposé dans la majorité des distributions Linux généralistes ces dernières années

- **Systemd** est un gestionnaire de systèmes et de services pour les systèmes d'exploitation Linux. Il est conçu pour être rétro-compatible avec **les scripts init**, et fournit un certain nombre de fonctionnalités:

  ⇒ Peut lancer **en parallèle** des processus interdépendants et **diminuer le nombre de processus** lancés au démarrage **--> Accélérer le démarrage du système**

  ⇒ **Rationaliser** la gestion des services

  ⇒ **Uniformiser la gestion des services** (architecture commune à toutes les distributions)

  ⇒ Inclure de **nouvelles fonctionnalités** dans le système d'init

  ⇒ …

Principe

- **Parallélisation des processus** par ouverture de **sockets** : **systemd** peut lancer en parallèle deux processus, même s'ils sont interdépendants.
- Pour chaque processus à lancer, **systemd** ouvre **une socket** unix.
- Les processus dépendant d'un processus donné vont se connecter à la socket correspondante, alors que le service réel n'est pas encore forcément actif.
- Lorsque le processus sera actif, **systemd** le connectera à sa socket et les clients pourront être servis.

Systemd: Organisation

- **systemd** (qui remplace **/sbin/init**) assure (via des fichiers de configuration) l'initialisation du système et la mise en place des services.
- **systemd** introduit la notion d'**unité**. **Une unité représente un fichier de configuration**. Une unité peut être **un service** (*.service), **un target** (*.target), **un montage** (*.mount), **un socket** (*.socket), ...

```
$ systemctl list-units
...
cups.socket     loaded active running   CUPS Printing Service Sockets
home.mount      loaded active mounted   /home
crond.service   loaded active running   Command Scheduler
sshd.service    loaded active running   OpenSSH server daemon
...
```


- Le répertoire **/usr/lib/systemd/system**, contient des fichiers de configuration. Chaque fichier décrit **une unité** et a un nom de la forme sshd.service, graphical.target, cups.socket, etc, suivant son type.

```
$ ls -l /usr/lib/systemd/system/cups*
-r--r--r--. 1 root root 126 20 nov.  16:07 /usr/lib/systemd/system/cups.path
-r--r--r--. 1 root root 198 20 nov.  16:07 /usr/lib/systemd/system/cups.service
-r--r--r--. 1 root root 131 20 nov.  16:07 /usr/lib/systemd/system/cups.socket
```

- Un autre exemple pour **le serveur ssh** :

```
$ rpm -ql openssh-server
...
/usr/lib/systemd/system/sshd-keygen.service
/usr/lib/systemd/system/sshd.service
/usr/lib/systemd/system/sshd.socket
...
```

Systemd: différents types d'unités

- **service :** lancement et administration de processus
- **socket :** lancement d'une socket de type Unix ou réseau, associée à un service.
- **target :** groupe fonctionnel de services et sockets. Généralise la notion de niveau d'éxécution.
- **mount :** gestion des montages, crées à la volée après lecture de /etc/fstab.
- **automount :** auto-montage par autofs.
- **Snapshots:** sauvegarder l'état actuel du système et de le restaurer ultérieurement.

Systemd: Fichiers de configuration

- Le répertoire **/usr/lib/systemd/system** est destiné aux unités système par défaut fournies par le système d'exploitation et les paquets installés,
- **/etc/systemd/system** est destiné aux unités système personnalisées ou modifiées par l'administrateur.

Systemd: commandes systemctl

```
# liste les unités
$ sudo systemctl list-units

# liste les services
$ sudo systemctl list-units --type=service
$ sudo systemctl --state=running

# contrôle d'un service
$ sudo systemctl start|stop|relaod|status|enable|disable <service>

# visualise le contenu d'un service
$ sudo systemctl cat <service>
```


Systemd: Cibles

- Sur les nouveaux systèmes, le concept des niveaux d'exécution a été remplacé par **les cibles (targets) Systemd.**
  - ⇒ Les cibles sont simplement des ensembles logiques d'unités.
  - ⇒ Il s'agit d'un type d'unité spécial **systemd** avec l'extension de fichier **.target**.
  - ⇒ Une cible **systemd** définit **l'état** dans lequel un système doit se trouver, ainsi que **les processus et services qui doivent être lancés** pour entrer dans cet état.

Niveaux d'exécution Systemd et init

SystemVinit Runlevel	Systemd Target	Notes
0	runlevel0.target, poweroff.target	Arrête le système
1, s, single	runlevel1.target, rescue.target	Mode single user.
2, 4	runlevel2.target, runlevel4.target, multi-user.target	Mode défini par l'utilisateur, identique au 3 par défaut.
3	runlevel3.target, multi-user.target	Multi-utilisateur, non grap
5	runlevel5.target, graphical.target	Multi-utilisateur, en mode
6	runlevel6.target, reboot.target	Redémarre

Les cibles (targets)

- Une cible (target) est un groupe fonctionnel d'unités.
- La cible **graphical** lance une interface graphique (**runlevel 5**).

/usr/lib/systemd/system/graphical.target

```
[Unit]
Description=Graphical Interface
Documentation=man :systemd.special(7)
Requires=multi-user.target
After=multi-user.target
Conflicts=rescue.target
AllowIsolate=yes

[Install]
Alias=default.target
```


Les cibles (targets)

- Une cible (target) est un groupe fonctionnel d'unités.
- La cible **graphical** lance une interface graphique (**runlevel 5**).

/usr/lib/systemd/system/graphical.target

```
[Unit]
Description=Graphical Interface
Documentation=man :systemd.special(7)
Requires=multi-user.target
After=multi-user.target
Conflicts=rescue.target
AllowIsolate=yes

[Install]
Alias=default.target
```

graphical.target requiert la cible **multi-user.target** et démarrera après que multi-user ait achevé son démarrage

Les cibles (targets)

- Une cible (target) est un groupe fonctionnel d'unités.
- La cible **graphical** lance une interface graphique (**runlevel 5**).

/usr/lib/systemd/system/graphical.target

```
[Unit]
Description=Graphical Interface
Documentation=man :systemd.special(7)
Requires=multi-user.target
After=multi-user.target
Conflicts=rescue.target
AllowIsolate=yes

[Install]
Alias=default.target
```

graphical.target ne peut pas être activée en même temps que la cible **rescue.target**

Gestionnaire de service Systemd

Les cibles (targets)

- Une cible (target) est un groupe fonctionnel d'unités.
- La cible **graphical** lance une interface graphique (**runlevel 5**).

/usr/lib/systemd/system/graphical.target

```
[Unit]
Description=Graphical Interface
Documentation=man :systemd.special(7)
Requires=multi-user.target
After=multi-user.target
Conflicts=rescue.target
AllowIsolate=yes

[Install]
Alias=default.target
```

La commande **initctl isolate graphical.target** passera la machine en mode graphique (le **init 5** du bon vieux temps)

System V init: résumé

- Système basé sur des **niveaux d'exécution** et une **série de scripts** lancés dans un ordre défini.
  - ⇒ **/etc/inittab** défini le niveau d'exécution
  - ⇒ **/etc/init.d/** contient les scripts écrit la plupart du temps en bash
  - ⇒ **/etc/rd[0-6].d/** contient des liens vers les scripts dont le nom donne l'ordre d'exécution.
- Le système souffrait de **plusieurs défauts** mais avait l'immense avantage d'être **simple et facile** à prendre en main

Systemd: résumé

- Les niveaux d'exécution SystemV sont remplacés par des target unit :
  - ⇒ **poweroff.target:** arrêt du système(runlevel 0)
  - ⇒ **rescue.target:** mode de secours (runlevel 1)
  - ⇒ **multi-user.target:** mode multi-utilisateur sans serveur graphique(runlevel 3)
  - ⇒ **graphical.target:** mode multi-utilisateur avec connexion graphique(runlevel 5)
  - ⇒ **reboot.target:** redémarrage du système (runlevel 6)

Autopsie d'un boot (1)

```
         Starting Replay Read-Ahead Data...
         Starting Collect Read-Ahead Data...
[OK] Reached target Login Prompts.
[OK] Listening on Syslog Socket.
[OK] Listening on /dev/initctl Compatibility Named Pipe.
[OK] Listening on Delayed Shutdown Socket.
[OK] Reached target Encrypted Volumes.
[OK] Set up automount Arbitrary Executable File Formats File System Automount Point.
         Expecting device dev-mapper-vg/x2dlv1.device...
[OK] Listening on udev Kernel Socket.
[OK] Listening on udev Control Socket.
         Expecting device dev-disk-by/x2duuid-039e27f3/x2d14d2.device...
         Expecting device dev-mapper-vg/x2dlv9.device...
[OK] Listening on Journal Socket.
[OK] Reached target Syslog.
         Starting Configure read-only root support...
         Mounting Media Directory...
         Starting Software RAID Monitor Takeover...
         Mounting Debug File System...
         Mounting POSIX Message Queue File System...
         Starting Journal Service...
[OK] Started Journal Service.
         Mounting Huge Pages File System...
         Starting udev Kernel Device Manager...
         Starting udev Coldplug all Devices...
[OK] Started Collect Read-Ahead Data.
[OK] Started Replay Read-Ahead Data.
```

Démarrage et arrêt du système Linux

Autopsie d'un boot (2)

```
[OK] Started Software RAID Monitor Takeover.
         Starting Load legacy module configuration...
         Starting Remount Root and Kernel File Systems...
         Starting Set Up Additional Binary Formats...
         Starting Apply Kernel Variables...
         Mounting Configuration File System...
         Starting Setup Virtual Console...
         Mounting Arbitrary Executable File Formats File System...
[OK] Started Apply Kernel Variables.
[OK] Mounted Media Directory.
[OK] Mounted Debug File System.
[OK] Mounted POSIX Message Queue File System.
[OK] Mounted Huge Pages File System.
[OK] Mounted Configuration File System.
[OK] Started udev Coldplug all Devices.
         Starting udev Wait for Complete Device Initialization...
[OK] Started Remount Root and Kernel File Systems.
[OK] Reached target Local File Systems (Pre).
[OK] Started udev Kernel Device Manager.
[OK] Started Configure read-only root support.
[OK] Started Setup Virtual Console.
[OK] Mounted Arbitrary Executable File Formats File System.
[OK] Started Set Up Additional Binary Formats.
[OK] Started Load legacy module configuration.
[OK] Found device /dev/mapper/vg-lv1.
         Activating swap /dev/mapper/vg-lv1...
[OK] Activated swap /dev/mapper/vg-lv1.
[OK] Reached target Swap.
```


Démarrage et arrêt du système Linux

Autopsie d'un boot (3)

```
[OK] Found device VIRTUAL_DISK.
         Starting File System Check on /dev/disk/by-uuid/039e27f3-14d2-436d-8f37-313a817b8c01...
[OK] Started udev Wait for Complete Device Initialization.
         Starting Wait for storage scan...
[OK] Started Wait for storage scan.
         Starting Initialize storage subsystems (RAID, LVM, etc.)...
systemd-fsck[667] : /dev/sda1 : recovering journal
systemd-fsck[667] : /dev/sda1 : clean, 352/51200 files, 96069/204800 blocks
[OK] Started File System Check on /dev/disk/by-uuid/039e27f3-14d2-436d-8f37-313a817b8c01.
         Mounting /boot...
[OK] Mounted /boot.
[OK] Found device /dev/mapper/vg-lv9.
         Starting File System Check on /dev/mapper/vg-lv9...
[OK] Started Initialize storage subsystems (RAID, LVM, etc.).
         Starting Initialize storage subsystems (RAID, LVM, etc.)...
systemd-fsck[713] : /dev/mapper/vg-lv9 : recovering journal
systemd-fsck[713] : /dev/mapper/vg-lv9 : clean, 11/1602496 files, 144597/6400000 blocks
[OK] Started File System Check on /dev/mapper/vg-lv9.
         Mounting /scratch...
[OK] Mounted /scratch.
         Starting Load Random Seed...
[OK] Listening on Device-mapper event daemon FIFOs.
         Starting Monitoring of LVM2 mirrors, snapshots etc. using dmeventd or progress polling...
[OK] Started Load Random Seed.
[OK] Started Monitoring of LVM2 mirrors, snapshots etc. using dmeventd or progress polling.
[OK] Reached target Local File Systems.
         Starting Tell Plymouth To Write Out Runtime Data...
         Starting Recreate Volatile Files and Directories...
[OK] Started Tell Plymouth To Write Out Runtime Data.
```


Démarrage et arrêt du système Linux

Autopsie d'un boot (4)

```
[OK] Listening on PC/SC Smart Card Daemon Activation Socket.
[OK] Listening on RPCbind Server Activation Socket.
[OK] Listening on CUPS Printing Service Sockets.
[OK] Listening on Avahi mDNS/DNS-SD Stack Activation Socket.
[OK] Listening on D-Bus System Message Bus Socket.
[OK] Reached target Sockets.
[OK] Reached target Basic System.
         Starting SYSV : cleantmp : clean /tmp and /var/tmp...
         Starting IPv4 firewall with iptables...
         Starting IPv6 firewall with ip6tables...
         Starting Console Mouse manager...
         Starting NTP client/server...
         Starting Kernel Samepage Merging...
         Starting Login Service...
         Starting Avahi mDNS/DNS-SD Stack...
         Starting System Logging Service...
[OK] Started System Logging Service.
         Starting D-Bus System Message Bus...
[OK] Started Restore Sound Card State.
[OK] Started Console System Startup Logging.
[OK] Started IPv4 firewall with iptables.
[OK] Started IPv6 firewall with ip6tables.
[OK] Started Console Mouse manager.
[OK] Started ACPI Event Daemon.
[OK] Started Machine Check Exception Logging Daemon.
[OK] Started SYSV : cleantmp : clean /tmp and /var/tmp.
[OK] Started Kernel Samepage Merging.
[OK] Started D-Bus System Message Bus.
[OK] Started NTP client/server.
```


Démarrage et arrêt du système Linux

Autopsie d'un boot (5)

```
         Starting OpenSSH server daemon...
         Starting Vsftpd ftp daemon...
         Starting Naming services LDAP client daemon....
         Starting Munin Node Server....
         Starting Automounts filesystems on demand...
         Starting Postfix Mail Transport Agent...
         Starting RPC bind service...
[OK] Started OpenSSH server daemon.
[OK] Started Vsftpd ftp daemon.
[OK] Started Naming services LDAP client daemon..
[OK] Started Automounts filesystems on demand.
[OK] Started RPC bind service.
         Starting NFS file locking service....
         Starting SYSV : Creates /tmp/.X11-unix/ if required...
[OK] Started SYSV : Creates /tmp/.X11-unix/ if required.
[OK] Started NFS file locking service..
[OK] Reached target Remote File Systems (Pre).
         Mounting /users...
[OK] Started Munin Node Server..
[OK] Started Postfix Mail Transport Agent.
[OK] Mounted /users.
[OK] Reached target Remote File Systems.
         Starting Permit User Sessions...
[OK] Started Permit User Sessions.
         Starting Job spooling tools...
[OK] Started Job spooling tools.
         Starting Wait for Plymouth Boot Screen to Quit...
         Starting Display Manager...
```

Arrêt du système (1)

- Les opérations de maintenance, diagnostics, modifications de logiciels, ajouts et retraits de matériel, taches administratives, coupures électriques, … nécessitent parfois **l'arrêt du système**.
- Cet arrêt peut être **planifié**, **périodique** ou **impromptu**
- Tous les systèmes Unix doivent être mis hors service correctement en utilisant les **commandes adéquates.**
- Ceci garantit **l'intégrité du disque** et la terminaison propre des différents services du système.

Arrêt du système (2)

- **Arrêt programmé** du système :
  - ⇒ utilisateurs prévenus de l'arrêt ;
  - ⇒ applications arrêtées proprement ;
  - ⇒ intégrité des systèmes de fichiers assurée ;
  - ⇒ sessions utilisateurs stoppées.
- **En fonction des options**, le système :
  - ⇒ passe en mode mono-utilisateur ;
  - ⇒ est arrêté ;
  - ⇒ est redémarré.
- **Commandes de mise hors service** :
  - ⇒ init ;
  - ⇒ shutdown ;
  - ⇒ halt ;
  - ⇒ reboot