ISET de Sfax
Département Technologie de l'Informatique

Administration Systèmes Linux
LPI 201/202

Chapitre 5

Gestion du noyau Linux
Installation et gestion des paquetages

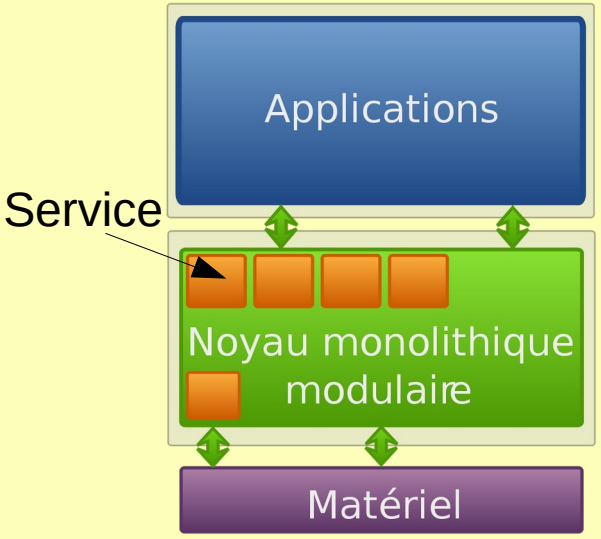
Enseignant : Azer ZAIRI

Courriel : Azer.ZAIRI@gnet.tn

Gestion du noyau Linux
Arrêt et démarrage du système

1. Introduction

- **Noyau** = **Kernel**
- Le noyau de Linux était créé par **Linus Torvalds** en **1991**
- Il est **opensource** et **gratuit** en licence **GPLv2** téléchargeable à partir de l'adresse : **https://www.kernel.org**
- Le noyau est la partie fondamentale de tous les systèmes d'exploitation.
- Le noyau :
  - gère les ressources de l'ordinateur
  - et permet aux différents composants
    - matériels
    - et logiciels
  - de communiquer entre eux.
- Les noyaux modernes de Linux sont monolithiques modulaires.
- **Monolithiques** : Seules les parties fondamentales du système sont regroupées dans un bloc de code unique
- **Modulaires** : Les autres fonctions, comme les pilotes matériels, sont regroupées en différents modules séparés



2. Les versions de Linux

- **Les numéros des versions Linux sont composés de la façon suivante (Exemple 5.4.0-91) :**
  - les deux premiers numéros, « **5.4** », indiquent le numéro principal du noyau
  - le troisième numéro **« 0 »** indique une révision. Autrement, c'est la version courante du noyau ;
  - Le quatrième chiffre **« 91 »** indique l'intégration des patchs de correction de bogues, de sécurité ou d'optimisation simples
- La version du noyau Linux utilisé, son nom et la version du compilateur utilisé sont indiqués dans le fichier **/proc/version**
  - cat /proc/version
- On peut aussi utiliser la commande **uname**
  - uname -a ou uname -r

Exemple :

- uname -a
- Linux LapTop-Azer 5.4.0-91-generic #102-Ubuntu SMP Fri Nov 5 16:31:28 UTC 2021 x86_64 x86_64 x86_64 GNU/Linux

3. Modification et configuration du noyau (1)

- **Le noyau Linux est diffusé avec une configuration générique** conçue pour supporter n'importe quelle application sur n'importe quel matériel.
- Cette configuration générique comprend de **nombreux pilotes de périphériques**, mais aussi des paramètres pour le noyau.
- On peut modifier ces paramètres afin d'**adapter le noyau à des besoins spécifiques**, augmenter les performances, renforcer la sécurité, ou encore la fiabilité du système.
- **Dans certains cas, la modification du noyau est nécessaire afin d'ajouter de nouveaux pilotes de périphériques.**
- Le code source du pilote ajouté doit être intégré dans les structures de données du noyau.
- Ceci peut exiger la re-compilation du noyau.

3. Modification et configuration du noyau (2)

- **Il existe quatre méthodes pour intervenir sur la configuration d'un noyau Linux :**
  - modification dynamique des paramètres de configuration du noyau ;
  - construction d'un noyau à partir de zéro (compilation du code source, avec éventuellement des modifications et des ajouts) ;
  - chargement de modules dans un noyau existant, à la volée ;
  - passage de paramètres en utilisant le chargeur de démarrage GRUB.
- Ces méthodes sont applicables dans des situations différentes.
- La modification dynamique des paramètres est la plus facile et la plus courante, tandis que la construction d'un noyau à partir des fichiers sources est la plus difficile et la moins souvent nécessaire.

4. La modification dynamique des paramètres du noyau

- **Le noyau peut être ajusté dynamiquement à travers des paramètres du système.**
- **Ces paramètres sont accessibles et modifiables à partir des fichiers du répertoire /proc/sys.**
  - **/proc/sys/fs/file-max** : Indique le nombre maximal de fichiers que le noyau peut manipuler simultanément
  - **/proc/sys/kernel/ctrl-alt-del** : Contrôle la gestion de la séquence Ctrl-Alt-Supp du clavier. S'il contient la valeur zéro, Ctrl-Alt-Supp est capturé et envoyé au programme init pour relancer le système correctement.
  - **/proc/sys/net/ipv4/icmp_echo_ignore_all** : Bloque les réponses au ping.
  - **/proc/sys/net/ipv4/ip_forward** : Active ou désactive le relayage (forwarding) entre les cartes réseaux. Activer le relayage est nécessaire pour faire fonctionner le système comme un routeur.
  - **/proc/sys/kernel/hostname** : Permet de changer le nom de la machine.

Remarque

- Puisque le système de fichiers **/proc** est **virtuel** et n'a d'existence qu'au niveau de la mémoire centrale, la modification des fichiers qu'il contient est temporaire, et sera perdue lors du redémarrage du système.
- Pour mémoriser les paramètres à appliquer au démarrage du système, on utilise le fichier **/etc/sysctl.conf**.

5. Construction d'un noyau Linux

- On peut être amené à compiler et installer un nouveau noyau Linux pour intégrer des correctifs (*patchs), des pilotes de périphériques ou de nouvelles fonctionnalités.*
- *Le* ***code*** **source** et les correctifs des noyaux Linux sont disponibles sur le site **www.kernel.org**.
- On doit bien étudier les besoins et les risques lors de la planification des améliorations et des correctifs du noyau.
- Généralement la nouvelle version peut apporter des fonctionnalités attrayantes, mais elle peut ne pas être aussi stable que la version courante.
- Donc on ne doit opter pour mettre à jour le noyau ou d'appliquer les correctifs que lorsque des gains en terme de productivité (souvent mesurés en terme de fiabilité et de performance) dépassent l'effort et la perte de temps nécessaires pour effectuer ces mises à jour des modules.

5.1 Configuration du noyau

- Les informations sur la configuration du noyau sont stockées dans le fichier .config situé à la racine du répertoire source du noyau.
- Il est déconseillé de modifier ce fichier manuellement.
- Linux offre plusieurs outils permettant de configurer le noyau et d'écrire le résultat dans **le fichier .config**.
  - **make config** : programme en mode texte
  - **make menuconfig** : utilitaire en mode texte écrit avec ncurses
  - **make gconfig** : outil graphique basé sur GTK+
  - **make xconfig** : outil graphique basé sur Qt
  - **make defconfig :** outil permettant de récupérer les paramètres de configuration par défaut du noyau.
  - **make oldconfig** : outil permettant de récupérer les paramètres de configuration d'une ancienne version du noyau
  - **make olddefconfig** : outil permettant de récupérer les paramètres de configuration par défaut du noyau
  - **make localmodconfig** : outil permettant de récupérer les paramètres de configuration du noyau actuelle sur la machine

5.1.1. L'outil : make config

- L'outil le plus élémentaire est **make config.**
- Il s'exécute en **mode console.**
- L'utilisateur doit ensuite spécifier toutes les options de configuration.
- **make config** demande pour chaque fonction si elle doit ou non être activée, en proposant quatre choix possibles sous la forme
  - [Y / m / n /?] :
  - Y (« yes ») pour intégrer la fonction directement dans le noyau.
    - C'est le choix par défaut, il peut être sélectionné en appuyant simplement sur la touche [Entrée] ;
  - m (« module ») pour construire un module qui va être chargé de façon dynamique ;
  - n (« no ») pour ne pas activer la fonction ;
  - ? pour afficher un message décrivant la fonction.

5.2. Compilation du noyau

- Une fois que la configuration du noyau est créée, on peut compiler le noyau en utilisant la **commande make** :
- Cette commande a pour effet de compiler le noyau ainsi que tous les modules nécessaires à cette configuration.
- Lors de la compilation du noyau, chaque fichier source compilé est affiché individuellement, avec des messages d'avertissement ou d'erreur éventuels.
- Si la compilation du noyau se termine sans erreur, le **résultat est un fichier binaire : Le fichier image du noyau**
- Ce fichier binaire doit être installé avant qu'on puisse l'utiliser au démarrage de la machine.

5.3. Installation et démarrage du nouveau noyau

- Une fois que le noyau est sous forme d'un fichier binaire, ainsi que les modules que le noyau va utiliser de façon dynamique, il ne reste qu'à installer le nouveau noyau et de démarrer le système.
- Dans cette étape, à la différence des étapes précédentes, toutes les commandes doivent être exécutées avec les droits root.
- L'installation du nouveau noyau peut être réalisée soit en utilisant des scripts offerts par la distribution installée, soit de façon manuelle.

6. Les partitions de Linux (1)

- Une installation de base ne nécessite que deux partitions:
  - **la partition racine /**
  - **et une partition de swap (mémoire virtuelle).**
- Le kernel Linux est plus performant lorsqu'un swap est disponible
- il peut ainsi y déplacer des données non fréquemment utilisées et donc gagner en performance disque (puisque les caches disques sont allouées de la mémoire libre).
- On peut également subdiviser le système en plusieurs partitions.

6. Les partitions de Linux (2)

- Le fait de créer plusieurs partitions permet d'avoir plusieurs avantages :
  - **partition / séparée de /usr** : Aucun passage des données entre les utilisateurs et le root
  - **Une partition de démarrage /boot indépendante** : Possibilité de dépannage et de réinstallation du chargeur de démarrage sans toucher au reste du système
  - **performance** : p.ex. plusieurs périphériques swap distribués sur plusieurs partitions de disques-durs différents ;
  - **séparation des logs et des fils d'attente** (/var/log et /var/spool) sur un serveur de mail très chargé
  - **indépendance à la réinstallation ou au changement de distribution** : mettre /home sur une partition séparée
  - **limiter les quotas des disques** par utilisateurs et/ou par groupes

6. Les partitions de Linux (3)

- La performance est souvent un problème négligé, mais elle est essentielle.
- Un partitionnement bien réalisé peut contribuer à améliorer la performance globale du système.
- Il faut commencer par définir les tâches ou les applications que le système doit exécuter.
  - **/ peu modifié, données cachées** : peut être monté depuis un disque lent.
  - **/var varie beaucoup (en particulier /var/log)** : on a intérêt à le placer sur un disque rapide
  - **/usr disque rapide :** conseillé par exemple dans le cas d'un serveur d'application
- En effet, la rapidité d'un disque est à la fois facteur de :
  - **sa vitesse brute** (liée à la vitesse de rotation et à sa densité)
  - **et de la latence** (durée de déplacement des têtes).

6. Les partitions de Linux (4)

- Une fois qu'un système est installé, on peut modifier les partitions avec la commande **fdisk**, ou d'autres implémentations comme **cfdisk**.
- On initialise une partition de swap avec **mkswap**,
- et on initialise une partition d'un système de fichiers avec les diverses instances de **mkfs** (p.ex. mkfs.ext2 ou mke2fs, mkfs.ext3 ou mke2fs -j, etc)
- Une fois initialisé, une partition de swap est activée avec **swapon**.
- Un système de fichiers est rattaché à un répertoire existant de l'arborescence via **mount**.
- Il remplace le contenu éventuel du répertoire temporairement.
- Pour automatiser le montage des systèmes de fichiers et l'activation du swap, on doit modifie ***/etc/fstab.***

7. Configuration de démarrage

- **Chargeur de premier niveau** = Chargeur de démarrage
  = first stage boot loader
- Le chargeur de premier niveau peut être soit :
  - **Le MBR** (Master Boot Record) standard MS-DOS
  - **Paquetage MBR sous Debian**, livré par GNU/Linux
  - **ou LILO** (Linux LOder)
  - **ou GRUB** (GRand Unified Boot loader)
- Le MBR occupe les 2 premiers bloc du disque :
  - **un bloc de démarrage**
  - **et un bloc contenant la table des partitions.**
- La taille de bloc est usuellement de 512 octets.
- **Le chargeur de deuxième niveau** n'est nécessaire que si LILO ou GRUB est installé dans une partition indépendante (dite aussi partition démarrable) et non pas dans le MBR.
- Une partition démarrable ne peut être qu'une partition Linux, et pas de swap.
- Si LILO (ou GRUB) est installé dans le MBR, le démarrage est possible de n'importe quelle partition Linux ou non Linux (sauf swap), y compris les partitions logiques ou dans d'autres disques.

7. Le chargeur GRUB

- **GRUB** : **GR**and **U**nified **B**ootloader
- Il est aussi installé dans le **MBR** (Master Boot Record)
  - C'est l'enregistre principal de démarrage qui est localiser au 512 premiers octets se trouvant dans le cylindre 0, tête 0 et secteur 0.
  - Il contient :
    - la table des partitions
    - Et les étapes d'amorçage du système d'exploitation
- Pour changer les options de GRUB on peut :
  - Editer le fichier **/boot/grub/grub.conf**.
  - Ce fichier est lu au démarrage par le programme /sbin/grub-install

7.1 Le fichier /boot/grub/grub.conf

- Il existe deux types de sections pour le fichier /boot/grub/grub.conf
  - **Section générale**

    default : nom de l'image à charger automatiquement

    timeout : le délai d'attente de l'invite en secondes
  - **Section image**

    title : nom de l'image

    root : la partition qui contient le chargeur de démarrage (second stage loader) et la racine du système de fichiers

    kernel : chemin du noyau en partant de la racine définie avec l'option root

    ro : lecture seule

    initrd : chemin du « initial root disk »

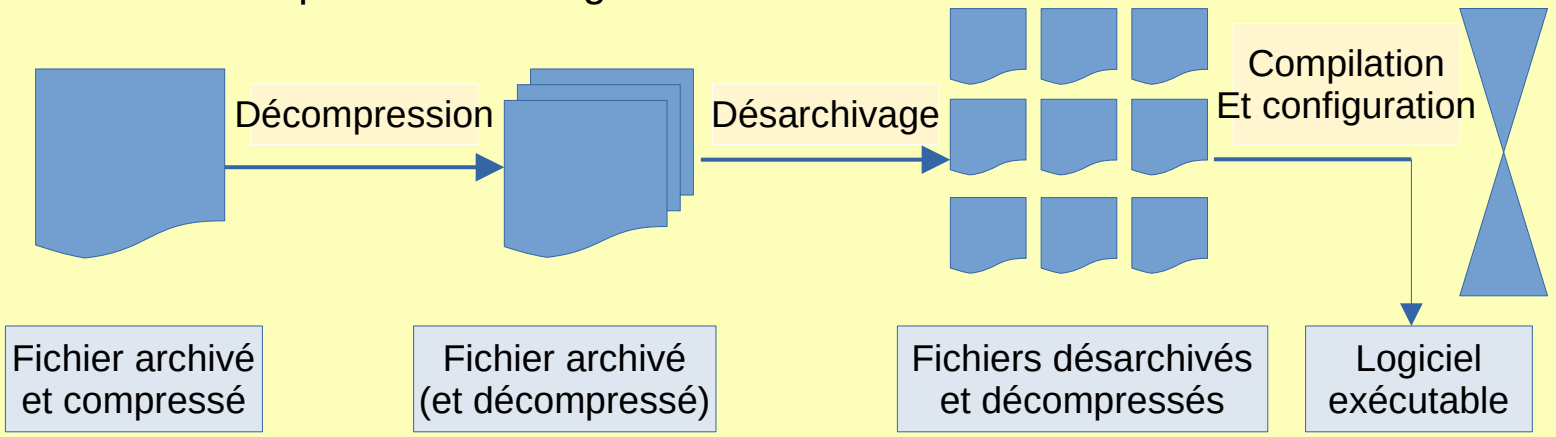
Installation et gestion des paquetages

1. Les méthodes d’installation

- **Les applications et les logiciels peuvent être installés de deux manières :**
  - Par la compilation des programmes sources
  - Ou par l’installation de paquetages

2. Installation à partir des sources

- **C’est la première et la plus ancienne méthode d’installation sous Linux**
- **Les sources sont composées d’un ou plusieurs fichiers archivés et compressés pour faciliter leur distribution**
- **Le format général des fichiers compressés et archivés est souvent :**
  - Nom_programme_version.tar.gz
- **Pour pouvoir installer ces types de fichiers sources il faut disposer des outils de :**
  - Décompression
  - Désarchivage
  - Compilation et configuration



2.1 Désarchivage et décompression (1)

- **La commande de décompression dépend de l'outil de compression avec lequel la compression a été réalisée.**
- **Il existe quatre outils de compression :**

Outils de compression	Commande de décompression	Extension du fichier
compress	uncompress	.Z
gzip	gunzip	.gz
bzip	bunzip	.bz
bzip2	bunzip2	.bz2

2.1 Désarchivage et décompression (2)

- **Une fois décompressé, le programme peut être désarchivé.**
- **L'outil d'archivage et de désarchivage classique des systèmes Linux s'appelle tar (tape archive).**
- **Tar est à la fois une commande d'archivage et de désarchivage**
- **Les principales options de la commande tar sont :**
  - c : pour créer l'archive
  - x : pour désarchiver
  - f : pour indiquer un fichier
  - v : pour donner des indications sur le déroulement du programme
- **Exemple : tar xvf nom_du_programme.tar**

2.1 Désarchivage et décompression (3)

- **Si l'archive original est un répertoire entier, toute la structure de son arborescence sera archivée**
- **Les versions récentes de la commande tar permettent la décompression et le désarchivage avec une seule commande en utilisant les options:**
  - z : pour la décompression à partir de gzip
  - j pour la décompression à partir de bzip2
- **Exemple**
- **tar xzvf nom_programme.tar.gz**
- **tar xjvf nom_programme.tar.bz2**

2.2 configuration

- Après la décompression et le désarchivage **il faut « configurer » le programme.**
- Généralement **il faut bien lire les instructions d'installation** qui se trouvent souvent dans un fichier texte avec le programme **(fichier README ou INSTALL)**
- La plupart des programmes peut générer automatiquement des fichiers qui facilitent la compilation du programme
- Ces fichiers sont appelés souvent **makefile**
- La génération du fichier makefile nécessite l'existence des outils **autoconf**
- Souvent un fichier script nommé **configure** existe à la racine du répertoire de l'application
- La commande **./configure** lancée à la racine du répertoire permet de lancer les outils **autoconf**
- Si le fichier « configure » n'existe pas on peut installer l'application avec la commande makefile

2.3 Compilation et installation

- La procédure d'installation est décrite dans un fichier texte se trouvant dans le répertoire du programme
- On peut souvent utiliser les commandes suivantes :
  - **make ou make all** : commande à lancer dans la racine du répertoire

    Remarque *: il est recommandé de* ***lancer les commandes make et ./configure avec un utilisateur normal***
  - **make install** : permet d'installer les programmes compilés sur le système.

    Cette commande nécessite d'être super-utilisateur, si les binaires seront installés dans /bin, /sbin, /usr ou /usr/local
- En cas de modification de source, il faut exécuter la commande **make clean** pour supprimer les fichiers précédemment générés par la compilation

3. Installation à partir des paquetages (1)

- C’est la deuxième méthode d’installation sur un système Linux.
- La plupart des distributions utilise un système de gestion de paquetages pour installer, désinstaller ou mettre à jour ses applications.
- **Avantages :**
  - Installation, désinstallation et mise à jour facile
  - Protection des fichiers de configuration
- **Inconvénients :**
  - Perte de performance due à la compilation sur une autre plateforme
  - Une corruption de la base des données des paquetages installés peut provoquer des problèmes de fonctionnement du système

3. Installation à partir des paquetages (2)

- **Les deux grandes familles d'outils de gestion des paquetages sont :**
  - RPM : Redhat Package Manager
  - DPKG : Debian PacKaGe

3.1 RPM (1)

- C’est un système **utilisé par les distributions compatible RedHat** (RedHat, Fedora,...)
- La gestion des paquetages est réalisée par la **commande « rpm »**
- Cette commande stocke sa base de données dans le répertoire **/var/lib/rpm.**
- Les programmes en format rpm ont souvent la structure du nom de fichier suivant :
  - Nom-version-release-architecture.rpm
- **Exemple :** blueeyes-1.41.1.ix64.rpm

3.1 RPM (2)

- **Voici les options courantes de la commande RPM**
  - **-i (ou –install)** : installe un paquetage
  - **-U (ou –update)** : met à jour un paquetage déjà installé ou l'installer s'il n'est pas encore installé
  - **-e (ou –erase)** : désinstalle un paquetage
  - **-q (ou –query)** : envoie une requête sur un paquetage afin d'afficher des informations pour savoir par exemple d'il est installé ou non
  - **-v (ou –verify)** : vérifie un paquetage
  - **-F (ou –freshen)** : met à jour un paquetage déjà installé
  - **--version** : affiche la version de la commande rpm
  - **--help** : affiche une aide sur la commande rpm

3.2 DPKG (1)

- C’est l’outil de **gestion de paquetage pour les distributions Debian** ou les autres distributions compatibles telle que **Ubuntu**
- Il permet d’installer, désinstaller, visualiser, configurer et construire des paquetages Debian
- **Les options courantes de la commande dpkg sont**
  - **-i** : installe l’application
  - **-r** : désinstalle l’application
  - **-l** : liste tous les paquetages installés
  - **-L** : liste les fichiers du paquetage indiqué
  - **--unpack** : désarchiver un paquetage sans le configurer
- La commande **dpkg-reconfigure** permet de reconfigurer un paquetage déjà installé

3.2 DPKG (2)

- DPKG est doté d'un autre outil de gestion avancé des paquetages appelé **APT (Advanced Package Tool)**
- APT utilise la commande dpkg en ajoutant des fonctionnalités supplémentaires :
  - La définition de la source des applications à installer (disques locaux, DVD, HTTP, HTTPS ou FTP)
  - La gestion des dépendances
- La définition des **sources des applications** se trouve dans le fichier **« /etc/apt/sources.list »**
- **Un dépôt de logiciels** se présente sous la forme d'une **ligne commençant par « deb »**,
  - ***Exemple :***
  - deb http://exemple.com/ubuntu/ exemple-version main (ou restricted)
- Un dépôt de code source (de logiciels) se présente sous la forme d'une ligne commençant par « deb-src »
  - ***Exemple :***
  - deb-src http://exemple.com/ubuntu/ exemple-version main (ou restricted)
- **main**: Paquet totalement libre | **restricted** : Paquet non libre

3.2 DPKG (3)

- **Les options de base de la commande apt sont :**
  - **apt-get update** : vérifier et mettre à jour le contenu et l'emplacement des fichiers sources indiqués dans le fichier /etc/apt/sources.list
  - **apt-get install nom-programme**: installe le paquetage indiqué
  - **apt-get remove nom-programme** : désinstalle le programme indiqué
  - **apt-get download nom-programme** : télécharger le programme (*.deb) sans l'installer
- **Remarques :**
  - **apt-get** est **équivalente** à **apt** et **aptitude**
  - Pour les distributions compatibles **RedHat** on utilise la commande **yum**

4. ALIEN : Outil de conversion entre paquetages

- L'outil **Alien** permet de réalises la conversion entre différents formats de paquets :
  - le format **.deb** : Pour les distributions compatibles **Debian**,
  - le format **.rpm** : Pour les distributions compatibles **Red Hat**
  - le format **.tgz** : Pour les distributions compatibles **Slackware**
  - le format **.slp** : Pour les distributions compatibles **Stampede**
  - le format **.pkg** : Pour les distributions compatibles **Solaris**
- **Exemples d'utilisation :**
  - Pour convertir un paquet .rpm en format .deb :
    - **sudo alien -d nom_du_paquet.rpm**
    - On aura donc un fichier "**nom_du_paquet.deb**"
  - Pour convertir un paquet .deb en format .rpm:
    - **sudo alien -r nom_du_paquet.deb**
    - On aura donc un fichier "**nom_du_paquet.rpm**"