Installation et gestion de paquetage

1.1 *Les principes d'installation*

Objectifs

- Comprendre les étapes de l'installation du système
- connaître les types d'installation courante

Points importants

Il existe plusieurs types d'installation du système d'exploitation Linux selon l'utilisation prévue du poste. Les bons choix effectués lors de l'installation facilitent la maintenance et la gestion d'un poste de travail.

Mots clefs

- type d'installation

a. Les étapes d'installation

En général, une installation du système d'exploitation Linux passe par les points suivants :

i. Démarrage de l'installation

L'installation peut être effectuée à partir du système local lui même (cdrom ou disquette) ou à partir du réseau (démarrer avec un système minimal à partir d'une disquette ou un cdrom qui permettra de configurer l'accès réseau. Les fichiers d'installation eux même se trouvent sur une autre machine du réseau : le serveur)

ii. Choix de la langue d'installation et le type de clavier

iii. Partitionnement du disque dur

Le schéma de partitionnement dépendra de la taille du disque ainsi que du type d'utilisation du poste.

Les outils de partitionnement intégré dans l'interface d'installation peuvent être **disk druid**, **fips**, **fdisk**, **sfdisk** ou **cfdisk** selon les distributions

iv. Choix des paquetages à installer

v. Configuration de l'interface graphique

Le choix des cartes graphiques, les gestionnaires de fenêtres et les paramétrages de l'affichage s'effectue à ce stade. L'installation de la souris s'effectue également dans cette rubrique.

vi. Configuration du réseau local

L'installation de(s) cartes(s) réseau(x) ainsi que les configurations réseau du poste se font à cette étape

vii. Choix du fuseau horaire

viii. Configuration de l'imprimante

ix. Définition du mot de passe du super utilisateur (root)

La création éventuelle des premières utilisateurs peut être effectuée lors de cette étape.

x. Création de la disquette de démarrage

xi. Configuration du chargeur de démarrage

b. Les types d'installation courante

Les distributions Linux actuelles offrent d'habitude trois options d'installation à l'utilisateur :

- option station de travail
- option serveur
- option installation personnalisée

Les fichiers d'installation peuvent se trouver sur un/des cdrom(s). Dans ce cas, il suffit de démarrer le système avec le premier cdrom d'installation. Des utilitaires de création de disquette de démarrage se trouvent souvent dans le premier cdrom : une image de la disquette de démarrage à copier sur une disquette avec :

- la commande **dd** sous Linux
- l'utilitaire **rawrite.exe** sous DOS ou MS Windows

Les fichiers d'installation peuvent se trouver également sur un autre ordinateur du réseau. Dans ce cas de figure, il faudrait lancer l'installation avec un cdrom ou une disquette contenant l'image **bootnet.img** avec les procédures citées ci-dessus.

Enfin, il est possible de créer un disque de récupération / réparation nommé « rescue disk ». Notons qu'au début du démarrage à partir d'un cdrom ou d'une disquette, l'interface d'installation permet soit

- une installation du système
- le mode « rescue » qui installe un mini système en mémoire afin de permettre une récupération / réparation d'un système existant

1.2 *Les chargeurs de démarrage : LILO / GRUB et le MBR*

Objectifs

- Comprendre le fonctionnement des chargeurs de démarrage
- connaître les deux types de chargeur
- Identifier les fichiers de configuration
- Savoir utiliser les fichiers de configuration

Points importants

Les deux principaux chargeurs de démarrage sont Lilo et Grub qui permettent de charger un des systèmes d'exploitation installés sur la machine.

Mots clefs

- lilo
- /etc/lilo.conf
- /boot/grub/grub.conf
- grub-install

Au démarrage d'un ordinateur, un programme de chargement situé dans le BIOS (Basic Input Output System) charge le MBR (Master Boot Record) qui est situé dans le premier secteur du disque. Ce MBR contient :

- un chargeur (loader) qui va à son tour chargé un autre programme chargeur (second stage loader) propre au système d'exploitation à charger.
- la description des tables de partitions du disque en cours
- une valeur numérique (magic number) utilisé parfois pour vérifier l'intégrité du MBR lui-même.

Les deux principaux chargeurs du système Linux sont LILO (Le Linux Loader) et GRUB (Grand Unified Bootloader), tous les deux installés dans le MBR.

Une fois lancé, LILO charge la deuxième étape du processus de chargement qui se situe généralement dans **/boot/boot.b**. Ce fichier permet également le démarrage d'autre système d'exploitation. Le fichier de configuration de LILO est **/etc/lilo.conf**. Ce fichier est lu à l'exécution de la commande **/sbin/lilo** qui crée à partir des configurations lues le fichier **/boot/map**. Ce dernier à son tour contient les emplacements physiques des blocs où se trouve le programme de démarrage étant de donné que le chargeur ne lit pas encore le système de fichiers à ce stade.

Le fichier **/etc/lilo.conf** est structuré de plusieurs lignes ayant la forme nom=valeur. Les principales options sont :

boot : localisation de lilo (dans le MBR ou dans à la tête d'une autre partition Linux, /dev/hda représente le MBR)
install : chemin d'accès du chargeur. La valeur par défaut de cette option est /boot/boot.b
prompt : permet à l'utilisateur d'effectuer un choix de système à charger avec un invite
default : nom de l'image à charger par défaut
timeout : utilisé avec l'option prompt, permet de définir un délai d'attente avant le démarrage automatique
image : chemin d'accès au noyau
label : nom de l'image pouvant être introduit par l'utilisateur si l'option prompt est activée
root : la partition qui contient le système de fichiers racine
read-only : pour monter la partition en lecture seule afin que le programme de vérification de l'intégrité du disque puisse s'executer proprement
append : permet de donner des paramètres aux modules du noyau
linear / lba32 : permet de commander LILO afin qu'il lise le disque en utilisant le système LBA (Linear Block Adressing).
other : permet d'indiquer la partition qui abrite un système d'exploitation autre que Linux

L'autre chargeur, GRUB est aussi installé dans le MBR. Pour changer les options de GRUB, on peut utiliser la commande **/sbin/grub** qui lancera un interpréteur de commande propre à GRUB. Les modifications seront enregistrées dans le fichier de configuration principale de GRUB, **/boot/grub/grub.conf** (qui peut être édité directement si besoin). Ce fichier de configuration est lu au démarrage par le programme **/sbin/grub-install**.

Il existe deux types de section pour le fichier **/boot/grub/grub.conf**

Section globale (global)

default : nom de l'image à charger automatiquement (la première entrée est identifiée par 0)
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 du système (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 »

1.3 *Les méthodes s'installation*

Objectifs

- Identifier les méthodes et les outils utilisés pour l'installation d'applications
- Installer une application à partir des sources
- Installer une application pré compilée

Points importants

Les applications ou logiciels peuvent être installés de deux façons : par la compilation des programmes sources ou par l'installation des paquetages.

Mots clefs

rpm
dpkg

A. Installation à partir des sources

La première et la plus ancienne méthode d'installation d'applications sur un système d'exploitation Linux est l'installation par les sources du programme. Les sources sont composées d'un ou plusieurs fichiers archivés et compressés pour faciliter le transport du programme. Leur forme est souvent **nom-du-programme-et-sa-version.tar.gz**. Pour pouvoir installer de telles applications, il faudrait donc avoir les outils de décompression, de désarchivage et de compilation installés sur le système.

a. désarchivage / décompression

La première étape de l'installation consiste alors à la décompression puis au désarchivage du fichier source.

La commande de décompression dépend de l'outil de compression avec laquelle la compression a été effectuée. Il existe actuellement trois outil de compression courant :

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

```
Ex : gunzip nom-du-programme.tar.gz
```

Une fois décompressé, le programme peut être désarchivé. L'outil d'archivage / désarchivage natif sur le système Linux s'appelle **tar** (tape archive). Tar est à la fois une commande d'archivage et de désarchivage. Notez les options :
-c pour créer l'archive
-x pour désarchiver
-f pour indiquer un fichier

-v pour donner des indications sur le déroulement du programme

```
Ex : tar -xvf nom-du-programme.tar
```

Si l'archive original était un répertoire entier, le répertoire et toute la structure de son arborescence seront recréés.

Notons aussi que les versions récentes de la commande **tar** permettent la décompression et le désarchivage avec une seule commande avec les options supplémentaires suivantes :

-z pour la compression à partir de gzip
-j pour la compression à partir de bzip2

```
Ex : tar -xzvf nom-du-programme.tar.gz
Ex : tar -xjvf nom-du-programme.tar.bz2
```

b. configuration

L'étape suivante consiste à explorer le programme. Il est recommandé de toujours lire les instructions relatives à l'installation de ce programme. Ces instructions d'installation se trouvent souvent dans le fichier README ou INSTALL ou TODO. D'autres fichiers du type README.plateforme ou INSTALL.plateforme qui correspondent à la distribution Linux utilisée ou au système d'exploitation utilisé peuvent exister selon les programmes.

La plupart des programmes peuvent générer des fichiers qui facilitent la compilation du programme. Ces fichiers sont communément appelés les « makefiles » dont la génération nécessite l'existence des outils **autoconf** sur le système. Les programmes **autoconf** inspectent le système ainsi que les applications déjà installées pour générer les fichiers makefiles. Souvent un script nommé configure existe à la racine du répertoire de votre application. La commande **./configure** lancée à la racine du répertoire de votre application permet de lancer les outils **autoconf**. Afin de bien utiliser ce script, il est conseillé de connaître les options qui peuvent être utilisées avec la commande **./configure --help**. Notez en particulier l'option **--prefix** qui permet de définir où l'application sera installée (la plupart des programmes utilise **/usr/local** comme prefix par défaut).

Enfin si le script configure est absent, l'application s'installe sans doute avec un fichier makefiles standard.

c. compilation / installation

La procédure d'installation est décrite dans un des fichiers d'instructions de l'installation du programme. Néanmoins, la suite la plus courante de commandes est la suivante :

make ou **make all** : commande à lancer dans la racine du répertoire du programme.

Remarque : il est recommandé de lancer toujours les commandes **./configure** et **make**

avec un utilisateur normale (sans être super utilisateur)

make install : permet d'installer les programmes compilés dans votre système. Cette commande nécessite d'être super utilisateur si les binaires seront installés dans **/bin** , **/sbin**, **/usr** ou **/usr/local**

Si les commandes précédentes se sont déroulées sans erreurs, l'application est maintenant installée et prête à être exécutée.

Pour refaire le processus, par exemple en cas de modification du source du programme, il faut auparavant faire **make clean** pour supprimer les fichiers précédemment générés par la compilation.

B. Installation avec les paquetages

La deuxième méthode d'installation d'applications sur un système Linux est l'installation avec les paquetages (pacages). La plupart des distributions utilisent un système de gestion de paquetages pour installer, désinstaller ou mettre à jour ses applications. Le tableau suivant présente les avantages et inconvénients de ces systèmes.

Avantages	Inconvénients
Installation et désinstallation facile	Perte de performance due à la compilation sur une autre plateforme
Mise à jour facile	
Protection des fichiers de configuration	
Gestion des paquetages installés facile	Une corruption de la base de données des paquetages installés peut rendre un système ingérable

Les deux grandes familles d'outils de gestion de paquetages sont RPM (Redhat Package Manager) et DPKG (Debian packages)

a. RPM

Système utilisé originalement par la distribution Redhat mais actuellement utilisé par la plupart des distributions.

La gestion des paquetages est principalement réalisée par la commande **rpm**. RPM garde sa base de données dans le répertoire **/var/lib/rpm.** Les programmes en format RPM porte souvent la structure du nom de fichier suivant :
nom-version-release-architecture.rpm

```
Ex : xsnow-1.41-1.i386.rpm
```

Ci jointes les options courantes :

-i (ou --install) : installe un paquetage

-U (ou --update) : met à jour un paquetage déjà installé ou installe si ceci n'est pas encore présent dans le système
-e (ou --erase) : désinstalle un paquetage
-q (ou --query) : envoie une requête sur un paquetage afin d'afficher des informations
-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 les options de la commande rpm

Options à utiliser avec l'option -q (ou --query)

c : affiche la liste des fichiers de configuration d'un paquetage donné
f : affiche le nom du paquetage auquel appartient un fichier donné
i : affiche les informations relatives à un paquetage
l : affiche tous les fichiers et repertoires relatifs à un paquetage
p : specifie que la requete est specifique au fichier du paquetage
b : pour créer un paquetage rpm à partir d'un repertoire contenant les fichiers sources
--rebuild : pour créer un paquetage à partir d'un fichier de source rpm
--requires PACKAGE :
--whatrequires CAPABILITY :

Les options spéciales

--nodeps : pour installer un paquetage sans se soucier des dépendances
--force : pour forcer la mise à jour
--test : faire un test (installation ou mise à jour) et afficher le resultat sur la sortie standard
--import : pour importer le fichier de signature d'un paquetage
--checksig : vérifie l'authenticité du paquet par sa signature

h : ajoute l'état d'avancement d'un processus en cours
v : mode bavard
a : appliquer l'option à tous les paquets installés

b. DPKG

Il s'agit de l'outil de gestion de paquetage pour la distribution Debian. Il permet d'installer, de désinstaller, de visualiser, de configurer, de construire des paquetages Debian. Les options courantes de la commande **dpkg** sont

-i nom-application.deb : Installe l'application nom-application.deb
-r monsoft.deb : Desinstalle l'application nom-application.deb
-l | grep appli : Cherche si le paquetage 'appli' est installé. Sans le grep, liste tous les packages.
-L nom-application : Liste les fichiers du paquetage 'nom-application' (s'il est installé) et leur emplacement.

DPKG est doté d'un autre outil de gestion avancé appelé **APT** (Advanced Package Tool). Apt utilise toujours la commande **dpkg** mais ajoute des fonctionnalités supplémentaires : la définitions de la source des applications à installer (disques local, cdrom ou sur Internet par le protocole http ou ftp) et la gestion des dépendances

La définition des sources des applications à installer s'effectue dans le fichier **/etc/apt/sources.list**. Ci joint un exemple des contenus de ce fichier

```
# See sources.list(5) for more information, especialy
# Remember that you can only use http, ftp or file URIs
# CDROMs are managed through the apt-cdrom tool.
deb http://http.us.debian.org/debian stable main contrib non-
free
deb http://non-us.debian.org/debian-non-US stable/non-US main
contrib non-free
deb http://security.debian.org stable/updates main contrib non-
free
```

apt-get update

apt-get va se connecter à tous les sites indiqués dans **/etc/apt/sources.list**, et va aller chercher les indexes de programmes disponibles

Les options de base de la commande apt sont :

apt-get install prog
    Installe le paquetage 'prog'
apt-get remove prog
    Désinstalle le paquetage 'prog'
apt-get --purge remove prog
    Désinstalle 'prog' ET ses fichiers de configuration
apt-get install prog1 prog2-
    Installe 'prog1' et désinstalle 'prog2'
apt-get remove prog1 prog2+
    Désinstalle 'prog1' et installe 'prog2'
apt-get --reinstall prog1
    Réinstalle le paquetage 'prog1'
apt-get update
    Met à jour la liste des packages disponibles.
apt-get upgrade
    Met à jour tous les packages pouvant être mis à jour.
apt-get -u upgrade
    Affiche en plus la liste des packages qui vont être mis à jour
apt-get dist-upgrade
    Met à jour le système entier (nouvelle version de la Debian)
apt-get source <prog>
    Télécharge la source de prog
apt-get -b source <prog>
    Télécharge le paquetage source et le compile ensuite.
apt-get build-dep <prog>
    Télécharge les dépendances du paquetage source <prog> qui va être compilé.

apt-get install prog/unstable

Installe 'prog' depuis la branche unstable

D'autres commandes de la famille apt existent. Elles permettent d'avoir des informations sur les paquetages. Ci jointes les options courantes de ces commandes.

apt-show-versions -u : Affiche une liste des packages pouvant être mis à jour.
apt-cache search <foobar> : Recherche dans la liste des packages disponibles les occurrences de <foobar>
apt-cache show <package> : Affiche la description de <package>
apt-cache depends <package> : Montre les dépendances de <package>
apt-file search <fichier> : Affiche le nom du package qui fournit <fichier>
apt-file list <package> : Affiche le contenu de <package>
apt-file maintient une base de données qui est mise à jour par apt-file update.

c. l'outil ALIEN

Le programme alien permet de changer un paquetage sous format rpm en format dpkg et vice versa.

1.4 *Gestion des bibliothèques*

Objectifs

- Gérer les librairies dynamiques

Points importants

Les bibliothèques constituent une place importante dans le bon fonctionnement du système Linux. Ce sont des programmes contenant des fonctions utilisées par d'autres programmes.

Mots clefs

/etc/ld/ld.conf
ldd
lddconfig
LD_LIBRARY_PATH

Les bibliothèques (libraries) sont des fonctions utilisées par par un programme binaire sous sa forme exécutable. Il y a deux sortes de bibliothèques : les bibliothèques statiques (qui sont inclus dans le fichier image de l'exécutable) ou les bibliothèques dynamiques ou

partagées (quand les codes du programme ne sont pas inclus dans le fichier image de l'exécutable). Les bibliothèques statiques se présentent sous forme de fichier avec l'extension **.a** tandis que les bibliothèques partagées ont l'extension **.so** (Shared Object). Les bibliothèques dynamiques peuvent être appelées par plusieurs programmes simultanément, et elles sont seulement associées au processus pendant l'exécution.

Les bibliothèques dynamiques sont chargées par l'utilitaire **ld.so** en respectant l'ordre de recherche des bibliothèques suivant :

- les répertoires mentionnés dans la variable d'environnement LD_LIBRARY_PATH
- le fichier cache /**etc/ld.so.cache** qui contient une liste de chemins de bibliothèques. Il est mis à jour par la commande **ldconfig** qui scrute les différents répertoires mentionnés dans le fichier **/etc/ld.so.conf**
- les répertoires **/lib** et **/usr/lib**

La commande **ldd** permet d'avoir la liste des bibliothèques partagées d'un exécutable.