02 - Maîtriser la configuration logicielle du système

Objectif

- Faire comprendre comment Linux installe, vérifie, relie et met à jour les logiciels de manière contrôlée.

1. RPM : rôle d'un gestionnaire de paquets

> Un paquet RPM n'est pas seulement une archive : c'est une unité d'installation, de traçabilité et de vérification.

1. RPM : rôle d'un gestionnaire de paquets

- Sur Red Hat, CentOS, Fedora et distributions proches, RPM sert de base à la gestion des paquets.
- Un éditeur peut bien distribuer une archive `.tar.gz`, mais un paquet RPM apporte une intégration système : installation, mise à jour, suppression, vérification, signatures, métadonnées et dépendances.

1. RPM : rôle d'un gestionnaire de paquets — À faire retenir

- Un paquet RPM permet de :
- installer, supprimer, mettre à niveau et vérifier un logiciel ;
- interroger une base locale des paquets installés ;
- transporter des métadonnées : nom, version, architecture, dépendances, scripts ;
- distribuer des paquets sources et binaires ;
- signer numériquement un paquet.

2. Structure d'un fichier RPM

> Un RPM contient de quoi identifier le paquet, prouver son intégrité, le décrire et installer sa charge utile.

2. Structure d'un fichier RPM

- Un fichier RPM est généralement composé de quatre grandes zones :

1. **Lead / identifiant** : permet de reconnaître le fichier comme RPM et contient des informations historiques de base.
2. **Signature** : permet de vérifier l'intégrité et parfois l'authenticité.
3. **En-tête** : contient les métadonnées modernes du paquet.
4. **Payload / charge utile** : contient les fichiers à installer, généralement compressés, puis stockés sous forme d'archive `cpio`.

2. Structure d'un fichier RPM — Démonstration possible

```
file paquet.rpm
rpm -qpi paquet.rpm
rpm -qpl paquet.rpm
rpm2cpio paquet.rpm | cpio -idmv
```

2. Structure d'un fichier RPM — Point d'attention

- La signature prouve que le paquet n'a pas été altéré et qu'il vient d'une source attendue, à condition que la chaîne de confiance soit correctement configurée.

3. Exécutables et bibliothèques

> Un binaire fonctionne rarement seul : il dépend souvent de bibliothèques partagées.

3. Exécutables et bibliothèques

- Lorsqu'on compile un programme, on transforme le code source en objets, puis on relie ces objets à des bibliothèques.
- Cette liaison peut être statique ou dynamique.
- **Liaison statique** : les bibliothèques nécessaires sont intégrées dans l'exécutable.
- **Liaison dynamique** : l'exécutable charge les bibliothèques au moment de l'exécution.
- Les bibliothèques partagées suivent souvent une convention de nommage :

```
libpthread.so.0
libc.so.6
```

- Le nom visible est souvent un lien symbolique vers une version précise.

3. Exécutables et bibliothèques — Où Linux cherche les bibliothèques

- Emplacements courants :

```
/lib
/lib64
/usr/lib
/usr/local/lib
```

- La configuration passe par :

```
/etc/ld.so.conf
/etc/ld.so.conf.d/
```

- La commande `ldconfig` construit un cache pour accélérer la résolution :

```
ldconfig
ldconfig -p
```

4. LD_LIBRARY_PATH

> `LD_LIBRARY_PATH` permet de forcer un chemin de bibliothèques, mais c'est un outil de contournement à manier avec prudence.

4. LD_LIBRARY_PATH

- On utilise `LD_LIBRARY_PATH` lorsqu'un programme doit charger une version particulière d'une bibliothèque avant celles du système.
- C'est utile pour tester une bibliothèque, exécuter un vieux logiciel ou lancer une application installée dans `/opt` avec ses propres dépendances.

```
export LD_LIBRARY_PATH=/usr/local/mylib:$LD_LIBRARY_PATH
./mon_programme
```

4. LD_LIBRARY_PATH — Point d'attention

- Ne pas en faire une configuration globale permanente sans raison : cela peut casser d'autres programmes en leur faisant charger de mauvaises bibliothèques.

5. Identifier les bibliothèques nécessaires

> `ldd` montre ce qu'un binaire va charger dynamiquement.

5. Identifier les bibliothèques nécessaires — Démonstration simple

```
ldd /usr/bin/ls
```

- Expliquer les lignes typiques :

```
libc.so.6 => /usr/lib/libc.so.6
/lib64/ld-linux-x86-64.so.2
```

- On voit à la fois les bibliothèques et le chargeur dynamique.

6. Construire un paquet RPM à partir des sources

> Empaqueter un logiciel permet de l'intégrer proprement au cycle de vie de la distribution.

6. Construire un paquet RPM à partir des sources

- Construire un RPM évite d'installer un logiciel « à la main » sans traçabilité.
- Le paquet peut ensuite être versionné, signé, placé dans un dépôt, installé et supprimé proprement.
- À présenter comme principe plutôt que comme procédure complète :

```
sources + spec file -> build -> RPM binaire + RPM source
```

7. Mettre en place un miroir local de paquets

> Un dépôt local réduit la dépendance réseau et accélère les installations.

7. Mettre en place un miroir local de paquets — Objectifs d'un dépôt local

- se passer temporairement d'un dépôt distant ;
- réduire la bande passante consommée ;
- accélérer installations et mises à jour ;
- proposer des paquets internes ou validés.

7. Mettre en place un miroir local de paquets

- En entreprise, un dépôt local est aussi un outil de gouvernance : on contrôle ce qui est disponible, quand les mises à jour arrivent, et quelles versions sont déployées.

8. Mises à jour système et patchs de sécurité

> Sur un serveur critique, on peut vouloir appliquer les correctifs de sécurité sans tout mettre à jour.

8. Mises à jour système et patchs de sécurité — Fedora

- Le cours présente la mise à niveau avec le plugin system-upgrade :

```
dnf install dnf-plugin-system-upgrade
dnf upgrade && dnf clean all
dnf system-upgrade download --releasever=<version>
dnf system-upgrade reboot
```

8. Mises à jour système et patchs de sécurité — CentOS / yum

- Lister les mises à jour de sécurité :

```
yum updateinfo list updates security
```

- Installer uniquement les mises à jour de sécurité :

```
yum update --security
```

8. Mises à jour système et patchs de sécurité — Point d'attention

- Cette sélection dépend des métadonnées de sécurité disponibles dans les dépôts.
- Si les dépôts ne fournissent pas ces informations, le filtrage sécurité devient incomplet.