01 - Installation avancée et déploiement

Objectif

- Montrer qu'une installation Linux en environnement professionnel doit être reproductible, automatisable et récupérable.

1. Automatiser l'installation avec Kickstart

> Kickstart transforme une installation interactive en scénario rejouable.

1. Automatiser l'installation avec Kickstart

- Lors d'une installation classique, un administrateur répond à une suite de questions : langue, clavier, réseau, partitions, mot de passe root, paquets, chargeur de démarrage.
- Kickstart permet de placer ces réponses dans un fichier texte.
- Une fois le système d'installation démarré, il lit ce fichier et déroule l'installation sans intervention humaine.
- Kickstart est historiquement associé à Red Hat, CentOS, Fedora et distributions proches.
- Côté Debian, l'équivalent est plutôt Preseed.
- Côté Ubuntu, on trouve des mécanismes dérivés ou compatibles selon les versions et outils utilisés.

1. Automatiser l'installation avec Kickstart

1. Créer un fichier Kickstart.
2. Le rendre accessible : support local, disque, NFS, HTTP, FTP ou HTTPS.
3. Préparer le média d'amorçage ou l'environnement PXE.
4. Rendre la source d'installation disponible.
5. Lancer l'installation automatisée.

1. Automatiser l'installation avec Kickstart

```
install
url --url=<url_du_depot>
lang en_US.UTF-8
keyboard us
network --onboot yes --device em0 --bootproto dhcp
rootpw <mot_de_passe>
reboot
selinux --enforcing
timezone Europe/Paris
bootloader --location=mbr
clearpart --all
%packages
@Base
@Core
%end
```

- Ce n'est pas la longueur du fichier qui compte, mais sa capacité à décrire clairement l'état final attendu de la machine.

1. Automatiser l'installation avec Kickstart — Validation

- Avant d'utiliser un fichier Kickstart, le valider avec `ksvalidator` :

```
yum install pykickstart
ksvalidator /path/to/kickstart.ks
```

1. Automatiser l'installation avec Kickstart — Point d'attention

- `ksvalidator` valide la syntaxe et certaines options, mais il ne garantit pas que l'installation réussira.
- Il ne valide pas réellement la logique des sections `%pre`, `%post` et `%packages`.

2. Kickstart via PXE

> PXE permet d'amorcer une machine par le réseau, Kickstart décrit ce qu'elle doit devenir.

2. Kickstart via PXE

- Dans un déploiement réseau, on retrouve souvent un serveur DHCP/BOOTP, un serveur TFTP ou HTTP, et une source d'installation accessible par NFS ou HTTP.
- Le fichier de configuration du chargeur de démarrage indique au noyau où trouver le fichier Kickstart.
- Exemple d'argument de boot :

```
inst.ks=http://<IP>/kickstarts/ks.cfg
```

- Avec GRUB ou GRUB2, l'idée reste la même : ajouter l'emplacement du fichier Kickstart à la ligne de boot.

3. Installation ROOT-on-LVM-on-RAID

> RAID protège contre la panne d'un disque, LVM apporte de la flexibilité d'allocation.

3. Installation ROOT-on-LVM-on-RAID

- LVM permet de ne plus raisonner uniquement en partitions fixes.
- On crée des volumes physiques, on les regroupe dans un groupe de volumes, puis on découpe ce groupe en volumes logiques.
- Les volumes logiques peuvent ensuite recevoir un système de fichiers.
- Dans un montage classique LVM sur RAID logiciel, on crée d'abord un volume RAID avec `mdadm`, puis on utilise ce volume RAID comme volume physique LVM.
- On parle donc de LVM au-dessus de RAID.

3. Installation ROOT-on-LVM-on-RAID — Exemple de montage

- Créer un RAID5 logiciel avec trois partitions :

```
mdadm --create /dev/md0 --level=5 --assume-clean --raid-devices=3 /dev/sd[bcd]1
```

- Créer ensuite la couche LVM :

```
pvcreate /dev/md0
pvdisplay /dev/md0

vgcreate data1 /dev/md0
vgdisplay data1

lvcreate -n Vol1 -L 4G data1
lvdisplay /dev/data1/Vol1
```

- Puis créer un système de fichiers sur le volume logique :

```
mkfs.ext4 /dev/data1/Vol1
```

3. Installation ROOT-on-LVM-on-RAID — Point d'attention

- La partition `/boot` ne doit pas être placée dans un volume logique si le chargeur d'amorçage ne sait pas la lire.
- Dans beaucoup d'architectures, on garde `/boot` séparé et simple.

4. Sécuriser le système de démarrage

> Si un attaquant contrôle le boot, il peut souvent contourner le système installé.

4. Sécuriser le système de démarrage

- La sécurité du démarrage vise surtout à se protéger contre un attaquant ayant un accès physique à la machine.
- Deux niveaux sont évoqués : le BIOS/UEFI et le chargeur de démarrage.
- Un mot de passe BIOS/UEFI peut empêcher la modification de l'ordre de boot ou le démarrage sur un support externe.
- Un mot de passe GRUB peut empêcher la modification temporaire des paramètres de boot et l'accès facile à un shell root via le mode mono-utilisateur.

4. Sécuriser le système de démarrage — GRUB2

- Définir un mot de passe GRUB2 peut se faire avec :

```
grub2-setpassword
```

- Le système crée alors un fichier contenant le hash du mot de passe.
- Selon les distributions, il faut aussi vérifier les entrées `menuentry` et retirer `--unrestricted` si l'on veut exiger le mot de passe pour démarrer une entrée précise.

4. Sécuriser le système de démarrage — Alternative si l'outil n'existe pas

```
grub-mkpasswd-pbkdf2
```

- Puis ajouter une configuration du type :

```
set superusers="admin"
password_pbkdf2 admin <hash_pbkdf2>
```

4. Sécuriser le système de démarrage — Point d'attention

- Une protection BIOS seule ne suffit pas : elle peut parfois être réinitialisée physiquement.
- Pour protéger réellement les données, il faut envisager le chiffrement disque.

5. Créer un média de recovery

> Un média de secours se prépare avant l'incident.

5. Créer un média de recovery

- Un live CD/DVD/USB permet de démarrer une machine quand le système installé est indisponible.
- Dans l'univers Fedora/Red Hat, l'outil `livecd-creator` peut créer une image à partir d'un fichier Kickstart.

```
yum install epel-release
yum install livecd-tools

livecd-creator --verbose \
  --config=/path/to/kickstart/file.ks \
  --fslabel=Image-Label \
  --cache=/var/cache/live
```

- Tester l'image avec QEMU/KVM :

```
qemu-system-x86_64 -m 2048 -vga qxl -cdrom filename.iso
```

- Écrire l'image sur une clé USB :

```
dd if=/path/to/image.iso of=/dev/sdX bs=8M status=progress oflag=direct
```

5. Créer un média de recovery — Point d'attention

- La commande `dd` est destructrice.
- Se tromper de périphérique peut effacer le mauvais disque.

6. Clonage d'une machine complète

> Cloner un disque, ce n'est pas seulement copier des fichiers : il faut aussi comprendre partitions, UUID, bootloader et systèmes de fichiers.

6. Clonage d'une machine complète

- Le clonage peut servir à sauvegarder, restaurer ou migrer une machine.
- Deux approches existent : copier les blocs avec `dd`, ou copier les fichiers avec `rsync`.
- La copie bloc conserve exactement la structure, mais transporte aussi les limites et anciens choix du système de fichiers.
- La copie fichier permet de recréer proprement le système de fichiers de destination.

6. Clonage d'une machine complète — Préparer les informations

```
parted -l > disk.bck
cat /etc/fstab > fstab.bck
```

- Copier une partition :

```
dd if=/dev/sda5 of=/mnt/backup_drive/sda5_root.img
```

- Réduire une image ext si nécessaire :

```
e2fsck -f sda5_root.img
resize2fs -M sda5_root.img
```

- Restaurer, puis réinstaller GRUB :

```
mount /dev/<root_destination> /mnt/root
chroot /mnt/root grub-install /dev/<disque>
```

6. Clonage d'une machine complète — Point d'attention

- Après une copie par blocs, les UUID sont conservés.
- C'est pratique pour `fstab`, mais dangereux si l'ancien et le nouveau disque coexistent sur la même machine.