R4B10 - Cryptographie et Sécurité

Unix

Bruno BEAUFILS

2024/2025

1. Utilisateurs

2. Fichiers

Utilisateurs/Groupes

- UNIX est un système **multi-utilisateurs**
  - accès aux fichiers propres à chaque utilisateur — `chmod(1)`, `chown(1)`
  - accès commun via des groupes — `chgrp(1)`, `newgrp(1)`
- identification des utilisateurs par
  - login / **uid**
  - groupe(s) / **gid**
- définition dans les bases système
  - *maps* : `passwd`, `group` — `getent(1)`
  - liaison entre symbole et identifiant — `id(1)`
  - définition des autres caractéristiques — `chsh(1)`, `chfn(1)`
- utilisateur particulier `root`
  - uid `0`
  - possède les accès à toutes les ressources
  - administrateur du système

Données utilisateurs

- servies par `PAM` pour
  1. authentification
  2. autorisation
  3. contrôle ouverture de sessions
- source par défaut = fichiers textes *(manipulables par les filtres standards)*
  - un enregistrement par ligne
  - champs séparés par des :
- `/etc/passwd` `passwd(5)`
  - définition utilisateurs
    1. login
    2. mot de passe chiffré
    3. uid
    4. gid
    5. informations complémentaires
    6. dossier principal
    7. chemin du shell
  - **fichier accessible en lecture à tout le monde**
    - raison historique
- `/etc/shadow` `shadow(5)`
  - déport des informations sensibles (mot de passe, etc.)
  - **fichier à accès limité** (`root` et groupe `shadow`)

1. Utilisateurs

2. Fichiers

Fichiers

Un **fichier** est une **suite linéaire d'octets**

- **abstraction logique** introduite par UNIX
  - différentes nature de fichiers
    - régulier, dossier, lien symbolique, tube nommé, socket, spécial
  - organisation **hiérarchique**
    - structure arborescente (via les dossiers)
    - une seule arborescence
    - **racine** de l'arbre : `/`
  - convention de nommage UNIX
    - FHS (*Filesystem Hierarchy Standard*) `hier(7)`
    - plus ou moins bien respecté (MacOS X, Fedora, NixOS, etc.)
- plusieurs localisations physique dans le **même** arbre
  - fichiers situés sur des périphériques différents
    - disque, partition, réseau

Contrôleur

- **organe électronique de commande des périphériques**
- plusieurs normes de communication entre ordinateur et périphériques
  - SCSI (interface parallèle), **SAS** (interface série)
  - PATA (interface parallèle), **SATA** (interface série)
  - **NVME**
    - dédiée disque disque SSD
    - utilise le bus PCI Express
- seule partie visible du noyau du système
  1. le noyau émet des requêtes de lecture/écriture de blocs
  2. le contrôler émet une interruption quand il a fini

Pilotes de périphériques *(device driver)*

- **programme de communication avec un contrôleur de périphérique**
- permet d'envoyer des commandes au périphérique
  - communique directement avec le contrôleur
  - permet de convertir les ordres UNIX en commandes appropriées
- une partie du noyau
  - langage spécifique au périphérique (et à l'interface)
  - identifié dans le noyau par un numéro spécifique unique
  - peut gérer plusieurs types de périphériques

Fichiers de périphériques

- **fichier spécial** (device file)
  - permet d'envoyer/recevoir des données à/de un périphérique
    - via les appels système fichier standard `read(2)`, `write(2)`
  - point d'accès à un pilote de périphérique (bout du noyau)
- accessibilité
  - dans `/dev/` `mknod(1)`, `udev(7)`
  - identification par 2 nombres (visibles avec `ls` ou `stat`)
    - nombre majeur *(major number)* = numéro du pilote
    - nombre mineur *(minor number)* = type de support dans le pilote
- 2 types
  - **caractère** *(character device)* : transfert octets par octets
  - **bloc** *(block device)* : tranferts par bloc d'octets
- désignation
  - `/usr/share/doc/linux-doc/Documentation/admin-guide/devices.txt`
  - exemples
    - `/dev/sda`, `/dev/sda1`
    - `/dev/nvme0`, `/dev/nvme0n1`, `/dev/nvme0n1p1`
    - `/dev/mmcblk0`, `/dev/mmcblk0p1`

Disque / Partition / Système de fichiers

- **disque** = périphérique de stockage de données
  - disque dur (HDD) (combinaison d'électronique, de mécanique et de magnétique)
  - disque SSD (électronique via mémoire flash)
  - clé USB ou carte mémoire (électronique via mémoire flash)
- **partition** = sous-ensemble d'un disque
  - intérêt :
    - séparation fonctionnelle : natures de données (système vs utilisateur)
    - séparation opérationnelle : types d'accès (`r`, `rw`)
    - contraintes opérationnelles : limite de taille
    - test (plusieurs OS sur la même machine par exemple)
  - commandes `fdisk(8)`, `sfdisk(8)`, `parted(8)`

```
apropos partition
```

- **système de fichiers** = organisation des blocs d'une partition
  - préparer = **formater** `mkfs(8)`, `fsck(8)`, `debugfs(8)`
  - format de stockage des blocs (*file system*
    - `ext2`, `ext3`, `ext4`, `ntfs`, `vfat`, `exfat`, `hfs`, `nfs`
  - offre des fonctionnalités spécifiques

Formatage

```
mkfs [-t TYPE] DEVICE
```

- wrapper vers les commandes ad-hoc
- la commande `mkfs -t ext4 /dev/sda1`
- exécute en fait `mkfs.ext4 /dev/sda1`

Accès aux systèmes de fichiers

- chaque système de fichier est limité à sa partition
- on peut **accrocher** un système de fichiers à un **répertoire**
  - accès aux fichiers via un chemin dans l'arbre
  - modification de la résolution des noms de fichiers
    - l'arborescence du système de fichiers est accessible dans le répertoire
  - le répertoire est le **point de montage**
    - doit exister
    - si non-vide : l'arborescence existante est recouverte (pas détruite)
- possibilité de spécifier les accès
  - lecture seule, lecture/écriture
  - changement de propriétaire
  - etc.

Montage / Démontage

```
mount [-t TYPE] [-o OPTIONS...] DEVICE NODE
umount NODE
umount DEVICE
```

- `DEVICE` fichier spécial d'accès à la partition
- `NODE` dossier point de montage
- `TYPE` nature du système de fichiers
- `OPTIONS`
  - exemples
    - `ro` / `rw`
    - `exec` / `noexec`
    - `sync` / `async`
    - `remount`
    - `defaults` = `rw,suid,dev,exec,auto,nouser,async`
  - associables par des virgules
  - chaque système de fichiers a des options spécifiques
- `mount(8)`, `umount(8)`

Montage automatique

- utilisation d'une **table de montage**
  - fichier texte `fstab(5)`
  - `/etc/fstab`
  - stocke les paramètres pour `mount`
  - un montage par ligne
- exemple

```
# /etc/fstab: static file system information.
#
# Use 'blkid' to print the universally unique identifier for a
# device; this may be used with UUID= as a more robust way to name devices
# that works even if disks are added and removed. See fstab(5).
#
# <file system> <mount point>   <type>  <options>         <dump>  <pass>
/dev/sda1       /               ext4    errors=remount-ro 0       1
/dev/nvme0n1p2  /boot           ext2    defaults          0       2
/dev/nvme0n1p1  /boot/efi       vfat    umask=0077        0       1
/dev/sda2       none            swap    sw                0       0
/dev/sda3       /usr            ext4    ro                0       3
/dev/sr0        /media/cdrom0   udf,iso9660 user,noauto   0       0
iut:/data       /home           nfs    defaults           0       0
```