Chapitre 12 : Gestion des utilisateurs

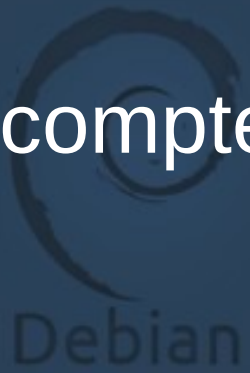
Comptes et protection des fichiers

Plan de ce chapitre

- Comptes
  - Définition, création (à connaître)
  - Protection des fichiers (à savoir)
  - Configuration des comptes

12.1 – Comptes et groupes

Utilisateur = compte + groupe(s)



Comptes Unix

- Un compte permet d'utiliser un ordinateur
  - On se connecte, on travaille, on se déconnecte
- Il en existe 3 sortes :
  - **Administrateur** : 1 seul possible, qualifié de superutilisateur, son nom de compte est root
    - Certains systèmes empêchent de s'y connecter directement car il a tous les droits. Voir plus loin.
  - **Comptes systèmes** : associés à certains services, ex : imprimante, réseau...
    - Non connectables, en général
  - **Utilisateurs ordinaires**

Groupes

- Les comptes sont placés dans des groupes
  - Exemple : étudiants, enseignants...
  - Exemple : groupes liés au système : bluetooth
- Un compte peut appartenir à plusieurs groupes
- Le fait d'appartenir à un groupe peut donner des droits sur des fichiers et des dossiers
  - Par exemple, si vous êtes dans le groupe cdrom, vous pouvez utiliser /dev/cdrom
  - Détails plus loin

Mémorisation des groupes

- Les groupes sont recensés dans `/etc/group`
- Le format : nom:x:n°:utilisateurs
  - Nom du groupe
  - Mot de passe du groupe : x = aucun
  - N° du groupe appelé GID
    - GID > 999 pour les utilisateurs normaux
  - Liste des utilisateurs qui en font partie
- Il y a 3 commandes pour gérer ce fichier (à moins de le faire à la main)

Création d'un groupe

- La commande addgroup ajoute un groupe :

  `addgroup` *`nom`*

  - L'option --gid GID permet de forcer le GID à une certaine valeur
  - Si on rajoute l'option --system on crée un groupe système (pour un service, voir prochain cours)

  Ex : `addgroup --gid 1001 developers`

- *Attention, il y a aussi la commande groupadd moins complète*

Modification d'un groupe

- On peut changer le nom d'un groupe

  `groupmod -n` *`nouveau ancien`*

  Ex : `groupmod -n programmers developers`

- On peut changer le GID d'un groupe

  `groupmod -g` *`GID groupe`*

  Ex : `groupmod -g 1002 programmers`

  Attention, si on change le GID, ça ne met pas à jour les fichiers et dossiers attribués au groupe => il va y avoir une anomalie, ces fichiers seront sans groupe connu (voir en TP)

Suppression d'un groupe

- La commande delgroup supprime le groupe

  `delgroup` *`nom`*

  Ex : `delgroup programmers`

- La commande est refusée s'il y a des utilisateurs dont c'est le groupe principal

Compte utilisateur

- Un compte est caractérisé par :
  - Un nom (logname), ex : lili
  - Un mot de passe (password)
  - Un numéro unique appelé UID, ex : 1001
  - Un home dir, ex : /home/lili
  - Un groupe principal identifié par le GID, ex : admin et des groupes secondaires, ex : cdrom, lpr, sudoer...
  - Un shell à lancer, ex : /bin/bash
  - D'autres informations appelées GECOS : le nom en clair, une adresse mail...

Mémorisation des comptes

- Les comptes sont définis dans le fichier `/etc/passwd`
  - Format : login:x:UID:GID:infos:homedir:shell
- Les mots de passe sont stockés dans le fichier `/etc/shadow`
  - Ce fichier est protégé, seul root peut le voir
  - Format : login:mdp:date:min:max
    - Date du mot de passe, durée minimum en jours et durée maximale en jour (vide ou 0 = pas de limite)

Authentification par PAM

- Les « Pluggable Authentication Modules » stockent les mots de passe sous forme cryptée dans `/etc/shadow`

  Ex : `$6$n4kRE2Hc$e09e6T00xUCgHzqgv0Zm1c`

- On ne peut pas les décrypter :
  - emploi d'algos non réversibles (sorte de projection)
- Quand on se connecte, le système PAM crypte le mot qu'on tape et le compare avec ce qui est dans /etc/shadow : égalité => OK

Piratage des mots de passe

- Pour « casser » un mot de passe, on emploie la force brute :
  - Passage en revue de tous les dictionnaires de mots, prénoms, dates (et leurs anagrammes)
  - Cryptage de chaque chaîne et comparaison avec le contenu de /etc/shadow
- Faisable si on travaille en « cloud computing »
  - Une multitude d'ordinateurs, chacun s'occupe de tester quelques mots

Ajout d'un utilisateur

- La commande adduser ajoute un utilisateur :

  `adduser --ingroup groupe nom`

  Ex : `adduser --ingroup programmers pierre`
- Elle demande les informations supplémentaires
- On peut forcer certaines caractéristiques :

  `--uid` *UID*

  `--shell` *shell*, ex : `--shell /bin/false`

  `--home` *homedir*

Affichage des informations

- La commande id affiche différentes informations :
  - UID et logname, GID et groupe principal, puis autres identifiants de groupes secondaires
  - Si on met un logname, ça affiche pour cet utilisateur
- `id -nG` seulement liste des noms de groupes auquel appartient l'utilisateur
  - La commande groups affiche la même liste

Modification d'un compte

- La commande usermod change ce qu'on veut avec les options :
  - `-g` *`groupe`* : change le groupe principal
  - `-c` *`infos`* : change les informations
  - `-u` *`UID`* : change l'UID de l'utilisateur
  - `-s` *`shell`* : change son shell de connexion
- Ex : `usermod -s /bin/bash pierre`

Appartenance à un groupe

- Tout utilisateur possède un groupe principal, celui qui est inscrit dans /etc/passwd
- On peut le rajouter dans d'autres groupes, dits secondaires :

  `usermod -G grpe2,grpe3... utilisateur`

  Ex : `usermod -G visiteurs,clients pierre`

  - Pour l'enlever d'un groupe, refaire la commande avec seulement les autres groupes
- Cela confère des privilèges supplémentaires

Changement de mot de passe

- La commande `passwd` permet de changer le mot de passe
  - `passwd` (sans paramètre) : change pour celui qui lance la commande
  - `sudo passwd` *`nomuti`* : change le mdp de cet utilisateur

Suppression d'un compte

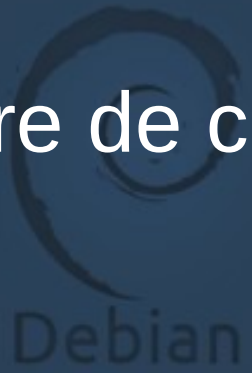
- La commande deluser permet de détruire un compte :

  ```
  deluser --remove-all-files nom
  ```

  - Sans l'option, la commande ne supprime pas les fichiers possédés par cet utilisateur

12.2 – Connexion sur un compte

Procédure de connexion



Connexion

- On peut se connecter à un compte (sauf celui de root sur Debian) lors de la mise en route du système
- Une fois connecté, on peut changer de compte à l'aide de la commande `su`
  - `su` : (sans paramètre) se connecter en tant que root
  - `su` *`utilisateur`* : se connecter en tant que cet utilisateur, mais rester dans le dossier courant
  - `su -` *`utilisateur`* : se connecter et aller dans le compte de cet utilisateur

Qui suis-je ?

- La commande `whoami` affiche le nom du compte dans lequel on travaille
- La commande `who` liste les utilisateurs connectés sur le même système

Lancer une commande en tant que superutilisateur

- La commande sudo permet de changer temporairement d'identité pour lancer une commande :
  - `sudo` *`commande`* : exécute la commande en tant que root
  - Il y a des options, mais généralement restreintes : il faut être connecté en tant que root
- En environnement graphique (Gnome sur Ubuntu), il faut faire gksudo au lieu de sudo si la commande ouvre une fenêtre.

12.3 – Protection des fichiers

Propriétaires et droits

Concepts

- Pour chaque fichier, on peut décider d'autoriser ou d'interdire les utilisateurs à y accéder :
  - Les utilisateurs : qui sont-ils par rapport au fichier ?
  - Le type d'accès : que veulent-ils faire avec ?
- Par exemple, je suis le propriétaire du fichier et je veux le modifier : puis-je ?
- Le système vérifie ces informations en permanence et bloque les commandes sur les fichiers qui ne sont pas autorisés

Les utilisateurs et les fichiers

- Un utilisateur = { un logname et des groupes }
- Un fichier = { un propriétaire (créateur) et un groupe (celui du propriétaire mais pas toujours) }
- Algorithme du système d'exploitation pour déterminer qui est cet utilisateur % ce fichier :
  - Si l'utilisateur = le propriétaire, alors il est dans la catégorie `U` (mal nommé, *user* au lieu de *owner*)
  - Sinon si l'un des groupes de l'utilisateur = celui du fichier alors il est dans la catégorie `G` (*group*)
  - Sinon, il est dans la catégorie `O` (*other*)

Les droits

- Si l'utilisateur veut, d'une manière ou d'une autre consulter ce qu'il y a à l'intérieur du fichier, alors il a besoin du droit R (*read*)
- S'il tente de modifier d'une manière ou d'une autre le contenu, alors il a besoin du droit W (*write*)
- S'il tente d'exécuter le contenu en tant que programme, il a besoin du droit X (*eXecute*)
- Ces trois droits peuvent être cumulés

Exemples d'accès multiples

- `more toto.txt`
  - Lecture sur toto.txt : droit R sur toto.txt
- `cp toto.txt tutu.lst`
  - Lecture de toto.txt : R sur toto.txt
  - Création et écriture de tutu.lst : W sur tutu.lst
- `echo oui >> riri.txt`
  - Modification de riri.txt (pas de lecture) : W sur riri.txt
- Voir en TP et TD pour se faire une idée

Mise en place des protections

- Chaque fichier possède un tableau de 3x3 booléens qui disent :
  - U possède tels droits parmi { R, W, X }
  - G possède tels droits parmi { R, W, X }
  - O possède tels droits parmi { R, W, X }
- Exemple :

	R	W	X
U	oui	oui	oui
G	oui	non	oui
O	non	non	oui

Exemple

- Soit un fichier titi.txt appartenant à jean, et ce fichier est placé dans le groupe clients
  - Si c'est jean qui veut l'afficher (R), alors c'est oui
  - Si c'est pierre du groupe clients qui veut l'afficher, alors c'est oui, car il y a R pour la catégorie G
  - Si c'est paul du groupe visiteurs qui veut l'afficher, alors c'est non

titi.txt	R	W	X
U	oui	oui	oui
G	oui	non	oui
O	non	non	oui

Affichage des droits

- La commande `ls -l` affiche le tableau des droits juste après le type du fichier
  - 3 triplets de 3 lettres, pour U, pour G et pour O
    1 lettre = ce droit accordé, - = refus
  - Nom du propriétaire et nom du groupe du fichier
- Exemple :

```
drwxrwxr-x 5 pierre admin   4096 nov. 25 bin
-rw-r--r-- 1 pierre admin    223 août 15 infos
-rwxr-x--x 2 pierre clients 4096 nov. 13 chk
  U   G   O
```

Protection des dossiers

- Le principe de la protection des dossiers est identique :
  - Les dossiers ont un propriétaire et un groupe
  - On compare l'utilisateur et le propriétaire, et éventuellement le groupe du dossier et les groupes de l'utilisateur
  - On regarde ce que l'utilisateur veut faire avec
    - Pb : ce sont les mêmes lettres RWX, mais pas tout à fait le même sens

=> c'est OK ou refusé.

Signification des droits

- Sur un dossier :
  - R signifie le droit de faire ls (sans être dedans)
    - Peu importe comment, par exemple more *.c fait un ls implicitement, car il a besoin de la liste pour filtrer les .c
  - W signifie le droit de créer un fichier, de renommer un fichier, de supprimer un fichier
    - Peu importe comment, par exemple `cp ../titi.txt .` exige W sur .
  - X signifie le droit de mentionner le dossier dans un chemin, d'aller dedans

Exemples d'accès

- `more tp1/toto.txt`
  - Entrée dans tp1 : droit X sur tp1
  - Lecture sur toto.txt : droit R sur toto.txt
- `cp tp2/*.txt tp3`
  - tp2 dans un chemin : droit X sur tp2
  - *.txt => liste du dossier tp2 : droit R sur tp2
  - Copie des .txt : droits R sur tous les .txt de tp2
  - tp3 dans un chemin : droit X sur tp3
  - Copie dans tp3 => créations : droit W sur tp3

Mise en place des droits

- La commande chmod permet d'ajouter des droits :

  `chmod catégories+droits noms`

  Ajoute les droits pour les catégories indiquées

  - Catégories = une ou plusieurs lettres parmi {u,g,o}
  - Droits = une ou plusieurs lettres parmi {r,w,x}
- Exemples
  - `chmod ug+x prog.sh`
  - `chmod go+r infos.txt`

Autres possibilités

- On peut définir les droits directement avec =

  `chmod catégories=droits noms`

  Ex : `chmod go=rx prog.sh`

- On peut retirer des droits avec -

  `chmod catégories-droits noms`

  Ex : `chmod o-r infos.txt`

- On peut mettre plusieurs spécifs à la suite :

  Ex : `chmod u=rw,g=r,o= docs.txt`

Variante de chmod

- Il existe une antique notation pour les droits, elle fait appel à l'octal (base 8)

  `chmod code8 fichiers...`

  code8 = 3 chiffres de 0 à 7 : pour U, pour G et pour O

  R = 4, W = 2, X = 1 => RW = 6, RX = 5, RWX = 7

  - Ex : `chmod 764 infos.txt`

    équivalent à faire : chmod u=rwx,g=rw,o=r infos.txt
- Défaut : ça bloque tout progrès du système des protections (ce qui manque beaucoup à Unix)

Droits par défaut

- Les droits par défaut (= ceux qui sont mis quand on crée un fichier) sont définis par la commande

  `umask` *`code8`*

  - Les nouveaux fichiers seront créés avec RW moins les droits indiqués dans le code8 (R=4, W=2, X=1)
  - Ex : `umask 027`

    Crée les fichiers avec les droits `RW- R-- ---`
  - NB : un fichier n'est jamais créé avec le droit X
- Pareil pour les dossiers, sauf : RWX moins les droits indiqués dans le code

Changer le propriétaire

- La commande chown permet de changer le propriétaire d'un fichier ou d'un dossier

```
chown proprio fichiers ou dossiers...
```

  - Ex : `sudo chown pierre /var/www/*.html`
- Elle ne peut être faite que par le superutilisateur
- Si on rajoute l'option -R, elle s'applique récursivement à tout le contenu d'un dossier
  - Ex : `sudo chown -R pierre /var/www`

Changer le groupe

- La commande chgrp permet de changer le groupe d'un fichier : on peut mettre un fichier dans un autre groupe que son propriétaire :

```
chgrp groupe fichiers ou dossiers...
```

  - Ex : `sudo chgrp visiteurs /var/www/*.html`
  - Elle ne peut être faite que par root (sudo)
- L'option -R permet d'appliquer à tout un dossier
  - Ex : `sudo chgrp -R visiteurs /var/www`

Changer propriétaire et groupe

- Pour faire les deux d'un coup :

```
chown proprio:groupe fichiers...
```

  - Ex : `sudo chown -R pierre:visiteurs /var/www`

Droits spéciaux sur les fichiers

- Il existe des droits spéciaux qu'on place sur certains fichiers (commandes d'administration)
  - Droit « SetUID » : faire `chmod u+s` *`programme`*
    - Il fait en sorte que, quand on lance le programme, on prenne temporairement l'identité (et les privilèges) du propriétaire du programme
    - Exemple : la commande /bin/passwd (pas sur DSL)
  - Droit « SetGID » : faire `chmod g+s` *`programme`*
    - Quand on lance ce programme, on passe temporairement dans le groupe du programme

Droits spéciaux sur les dossiers

- Sur les dossiers
  - Droit « SetGID » : faire `chmod g+s` *`dossier`*
    - Normalement, quand un utilisateur crée un fichier dans un dossier, ce fichier est mis dans le groupe de cet utilisateur. Sauf quand il y a le SetGID sur le dossier : le fichier créé est mis dans le même groupe que le dossier.