Administration système: Création des utilisateurs et des groupes

Dr. Ghada Jaber

Maître de conférences

Laboratoire Heudiasyc

2022/2023

Plan

Introduction

- La gestion des utilisateurs et des groupes sur une machine Linux est l'un des aspects clés de l'administration système
- Linux est un système d'exploitation multi-utilisateurs dans lequel plusieurs utilisateurs peuvent utiliser la même machine en même temps
- Les informations sur les utilisateurs et les groupes sont stockées dans quatre fichiers dans l'arborescence du répertoire ***/etc/*** :
  - ***/etc/passwd*** un fichier de sept champs délimités par des deux points, contenant des informations de base sur les utilisateurs
  - ***/etc/group*** un fichier de quatre champs délimités par des deux points, contenant des informations de base sur les groupes
  - ***/etc/shadow*** un fichier de neuf champs délimités par des deux points, contenant les mots de passe hachés des utilisateurs
  - ***/etc/gshadow*** un fichier de quatre champs délimités par des deux points, contenant des mots de passe de groupe hachés

Plan

I. Introduction
II. Fichiers
III. Ajout et suppression des comptes utilisateurs
IV. Ajout et Suppression des groupes
V. Gestion des propriétés et des droits d'accès aux fichiers
VI. Répertoires et fichiers spéciaux

Fichiers

Fichier etc/passwd:

- **/etc/passwd** est un fichier lisible par tout le monde qui contient une liste d'utilisateurs, chacun sur une ligne séparée :

benoit:x :1001:1001: :/home/benoit:/bin/bash

(1) (2) (3) (4) (5) (6) (7)

(1) Nom d'utilisateur: Le nom utilisé lorsque l'utilisateur se connecte au système

(2) Mot de passe: Le mot de passe haché (ou un x si des mots de passe *shadow* sont utilisés)

(3) User ID (UID): Le numéro d'identification attribué à l'utilisateur dans le système

(4) Group ID (GID)Le numéro du groupe principal de l'utilisateur dans le système

(5) GECOS: Un champ de commentaire optionnel, qui est utilisé pour ajouter des informations supplémentaires sur l'utilisateur (comme le nom complet). Le champ peut contenir plusieurs entrées séparées par des virgules

(6) Répertoire personnel: Le chemin absolu du répertoire personnel de l'utilisateur

(7) Shell: Le chemin absolu du programme qui est automatiquement lancé lorsque l'utilisateur se connecte au système (généralement un shell interactif tel que /bin/bash)

Fichiers

Fichier /etc/group:

- **/etc/group** est un fichier lisible par tout le monde qui contient une liste de groupes, chacun sur une ligne séparée :

thedeveloper: x :1002 :

(1) (2) (3) (4)

(1) Nom du groupe

(2) Mot de passe: Le mot de passe haché du groupe (ou un x si des mots de passe *shadow* sont utilisés)

(3) Group ID (GID)Le numéro d'identification attribué au groupe dans le systèm

(4) Liste des membres: Une liste, délimitée par des virgules, des utilisateurs appartenant au groupe, à l'exception de ceux il s'agit du groupe principal

Fichiers

Fichier /etc/shadow:

- **/etc/shadow** est un fichier lisible par tout le monde qui contient une liste de groupes, chacun sur une ligne séparée :

joe:6i9gjM4Md4MuelZnj!khdfjhdfhj/odefjijfdokdofkfojofCd$/:18029:0:99999: 7 : : :
(1) (2) (3) (4) (5) (6) (7) (8) (9)

(1) Nom d'utilisateur
(2) Mot de passe haché
(3) Date du dernier changement du mot de passe: le nombre de jour écoulé depuis le 01/01/1970. Si la valeur de ce champ est égal à 0 ça veut dire que l'utilisateur doit changer son mot de passe lors du prochain accès
(4) Age minimum du mot de passe: Le nombre minimum de jours, après un changement de mot de passe, qui doit s'écouler avant que l'utilisateur soit autorisé à changer à nouveau le mot de passe
(5) Âge maximum du mot de passe: Le nombre maximum de jours qui doivent s'écouler avant qu'un changement de mot de passe soit nécessaire.
(6) Période d'avertissement du mot de passe: Le nombre de jours, avant l'expiration du mot de passe, pendant lesquels l'utilisateur est averti que le mot de passe doit être modifié
(7) Période d'inactivité du mot de passe: Le nombre de jours après l'expiration d'un mot de passe pendant lesquels l'utilisateur doit le mettre à jour. Après cette période, si l'utilisateur ne modifie pas le mot de passe, le compte sera désactivé
(8) Date d'expiration du compte: La date, en nombre de jours depuis le 01/01/1970, à laquelle le compte utilisateur sera désactivé. Un champ vide signifie que le compte utilisateur n'expirera jamais.
(9) Un champ réservé: Un champ qui est réservé pour un usage futur

Fichiers

Fichier /etc/gshadow:

- **/etc/gshadow** est un fichier lisible uniquement par root et par les utilisateurs disposant de privilèges root qui contient des mots de passe hachés par les groupes, chacun sur une ligne séparée:

developer:6i9gjM4Md4MuelZnjkhdfjhdfhofkfojofCd$/: :

(1) (2) (3) (4)

(1) Nom du groupe

(2) Mot de passe haché: Le mot de passe haché du groupe (il est utilisé lorsqu'un utilisateur, qui n'est pas membre du groupe, veut rejoindre le groupe en utilisant la commande newgrp si le mot de passe commence par ! personne n'est autorisé à accéder au groupe avec newgrp).

(3) Administrateurs du groupe: Une liste, délimitée par des virgules, des administrateurs du groupe (ils peuvent changer le mot de passe du groupe et peuvent ajouter ou supprimer des membres du groupe avec la commande gpasswd).

(4) Membres du groupe: Une liste des membres du groupe, délimitée par des virgules.

Plan

I. Introduction
II. Fichiers
III. Ajout et suppression des comptes utilisateurs
IV. Ajout et Suppression des groupes
V. Gestion des propriétés et des droits d'accès aux fichiers
VI. Répertoires et fichiers spéciaux

Ajout et suppression des comptes utilisateurs

- Les attributs qui caractérisent un utilisateur Unix sont :
  - un nom de connexion (login)
  - un mot de passe
  - un identifiant numérique unique (UID)
  - un groupe primaire (GID) ;
  - un commentaire (appelé gecos);
  - le répertoire principal de l'utilisateur (home directory)
  - un interpréteur de commandes (shell) par défaut

- Un utilisateur est identifié par le système par son UID
- L'utilisateur root a pour UID 0. C'est cette caractéristique qui lui confère un accès complet au système
- L'ensemble de ces éléments est stocké dans le fichier /etc/passwd au format texte
- Les champs sont séparés par le caractère « : »; Par exemple : jo:x:500:500:Jo Dalton:/home/jo:/bin/bash

Ajout et suppression des comptes utilisateurs

❖ Pour ajouter un utilisateur sous linux: **Useradd sr01**

```
root@ghada-HP-EliteBook-830-G6:/home/ghada# useradd sr01
root@ghada-HP-EliteBook-830-G6:/home/ghada#
```

Options:

-c: Crée un nouveau compte utilisateur avec des commentaires personnalisés (par exemple le nom complet)
-d: Crée un nouveau compte utilisateur avec un répertoire personnel défini
-e: Crée un nouveau compte utilisateur en fixant une date précise à laquelle il sera désactivé
-f: Crée un nouveau compte utilisateur en fixant le nombre de jours après l'expiration du mot de passe pendant lesquels l'utilisateur doit mettre à jour son mot de passe
-g: Crée un nouveau compte utilisateur avec un GID spécifique
-G: Crée un nouveau compte utilisateur en l'ajoutant à plusieurs groupes secondaires
-m: Crée un nouveau compte utilisateur avec son répertoire personnel
-M: Crée un nouveau compte utilisateur sans son répertoire personnel
-s: Crée un nouveau compte utilisateur avec un shell de connexion spécifique
-u: Créer un nouveau compte utilisateur avec un UID spécifique

Ajout et suppression des comptes utilisateurs

- Une fois le nouveau compte utilisateur créé, vous pouvez utiliser les commandes id et groups pour connaître son UID, son GID et les groupes auxquels il appartient.

```
root@ghada-HP-EliteBook-830-G6:/home/ghada# id sr01
uid=1002(sr01) gid=1002(sr01) groupes=1002(sr01)
root@ghada-HP-EliteBook-830-G6:/home/ghada#
```

```
root@ghada-HP-EliteBook-830-G6:/home/ghada# groups sr01
sr01 : sr01
root@ghada-HP-EliteBook-830-G6:/home/ghada#
```

- Pour supprimer un utilisateur **Userdel sr01**

- Après avoir créé le nouvel utilisateur, vous pouvez définir un mot de passe à l'aide de passwd :

```
root@ghada-HP-EliteBook-830-G6:/home/ghada# passwd sr01
Entrez le nouveau mot de passe UNIX :
Retapez le nouveau mot de passe UNIX :
passwd : le mot de passe a été mis à jour avec succès
root@ghada-HP-EliteBook-830-G6:/home/ghada#
```

- Pour filtrer les bases de données de mot de passe et de groupe, vous pouvez utiliser la commande grep:
  Cat /etc/passwd | grep sr01 ou grep sr01 /etc/passwd

Ajout et suppression des comptes utilisateurs

Repertoire Squelette:

- Lorsque vous ajoutez un nouveau compte utilisateur, même en créant son répertoire personnel, le répertoire personnel nouvellement créé est rempli de fichiers et de dossiers qui sont copiés à partir du répertoire squelette (par défaut /etc/skel).

- Le répertoire squelette contient les fichiers et répertoires qui seront copiés dans le répertoire personnel de l'utilisateur au moment de sa création.
- **ls -la /etc/skel/**

```
root@ghada-HP-EliteBook-830-G6:/home/ghada# ls -la /etc/skel
total 40
drwxr-xr-x   2 root root  4096 août   5  2019 .
drwxr-xr-x 130 root root 12288 déc.   9  2022 ..
-rw-r--r--   1 root root   220 avril  4  2018 .bash_logout
-rw-r--r--   1 root root  3771 avril  4  2018 .bashrc
-rw-r--r--   1 root root  8980 avril 16  2018 examples.desktop
-rw-r--r--   1 root root   807 avril  4  2018 .profile
```

Ajout et suppression des comptes utilisateurs

- Passwd est utilisée pour changer le mot de passe d'un utilisateur. Tout utilisateur peut changer son mot de passe, mais seul root peut changer le mot de passe de n'importe quel utilisateur
- Selon l'option de passwd utilisée, vous pouvez contrôler des aspects spécifiques du vieillissement des mots de passe :
  - **-d:** Supprime le mot de passe d'un compte utilisateur (ce qui désactive l'utilisateur)
  - **-e:** Force le compte utilisateur à changer le mot de passe
  - **-l:** Verrouille le compte de l'utilisateur (le mot de passe haché est précédé d'un point d'exclamation)
  - **-u:** Déverrouille le compte utilisateur (il supprime le point d'exclamation)
  - **-S:** Produire des informations sur le statut du mot de passe pour un compte spécifique

- Il est donc possible de verrouiller un compte utilisateur

Plan

I. Introduction
II. Fichiers
III. Ajout et suppression des comptes utilisateurs
IV. Ajout et Suppression des groupes
V. Gestion des propriétés et des droits d'accès aux fichiers
VI. Répertoires et fichiers spéciaux

Ajout et suppression des groupes

- Les attributs qui caractérisent un groupe Unix sont :
  - un nom
  - un mot de passe (jamais utilisé)
  - un identifiant numérique unique (GID)
  - une liste d'utilisateurs membres

- La liste des utilisateurs peut être vide ou contenir un plusieurs nom d'utilisateurs séparés par un caractère « , »

- L'ensemble de ces éléments est stocké dans le fichier /etc/group au format texte
- Les champs sont séparés par le caractère « : ». Par exemple : sr01:x:100:jo,jack,william,bastien

Ajout et suppression des groupes

- Vous pouvez ajouter ou supprimer des groupes à l'aide des commandes **groupadd** et **grouppdel**
- L'option -g de cette commande crée un groupe avec un GID spécifique.
- Si vous souhaitez supprimer le groupe developer, vous pouvez lancer la commande suivante : # **groupdel sr01g1**
- On peut ajouter un utilisateur à un groupe avec **gpasswd** :

```
root@ghada-HP-EliteBook-830-G6:/home/ghada# groupadd -g 1090 sr01g1
root@ghada-HP-EliteBook-830-G6:/home/ghada# gpasswd -a sr01 sr01g1
Ajout de l'utilisateur sr01 au groupe sr01g1
root@ghada-HP-EliteBook-830-G6:/home/ghada#
```

- On change les paramètres des groupes avec le programme usermod:
  - **-d:** répertoire utilisateur
  - **-g:** définit le GID principal
  - **-l:** identifiant utilisateur
  - **-u:** UID utilisateur
  - **-s:** shell par défaut
  - **-G:** ajoute l'utilisateur à des groupes secondaires
  - **-m:** déplace le contenu du répertoire personnel vers le nouvel emplacement

Ajout et suppression des groupes

- Pour modifier les paramètres d'un groupe, on peut utiliser commande **groupmod**

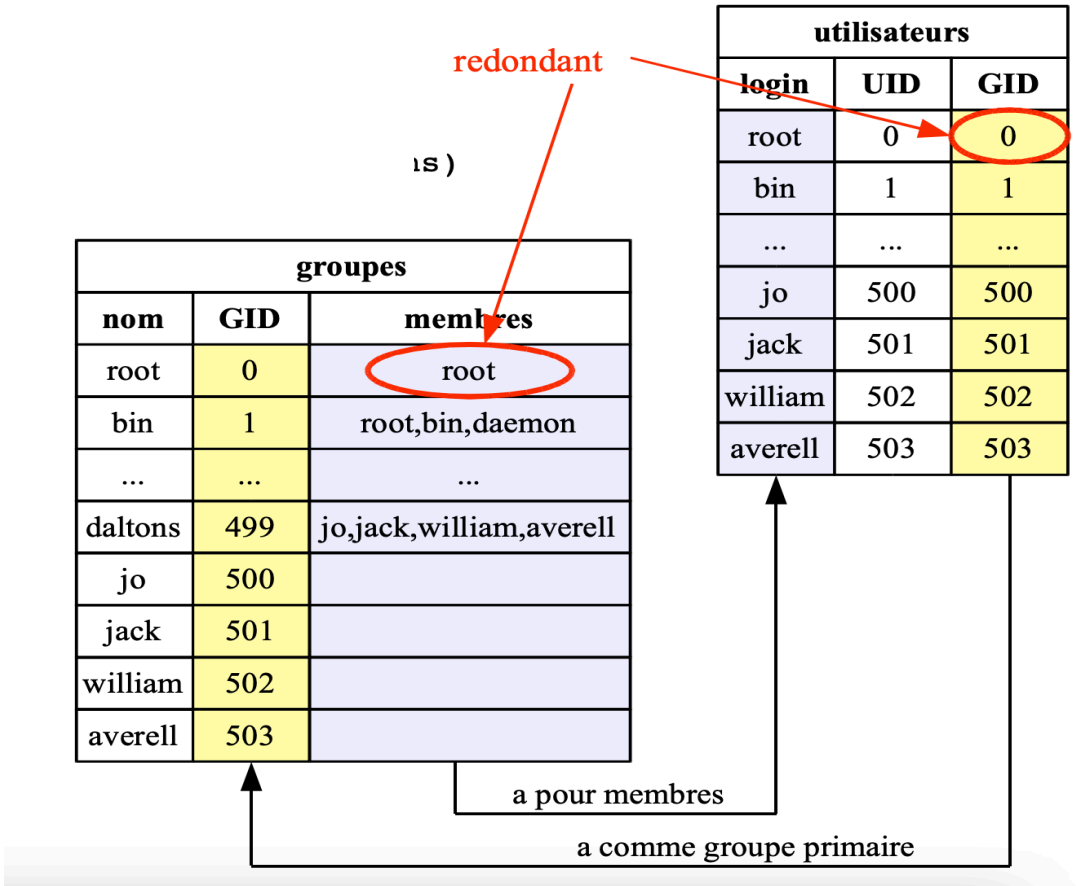
Options:

-g: GID

-n: nom du groupe

- La commande **id** permet de lister les informations (UID, GID, groupes) relatives aux utilisateurs.

```
$ id jo
uid=500(jo) gid=500(jo) groupes=500(jo),499(daltons)
```



Ajout et suppression des groupes

Conventions

❖ En général, la liste des utilisateurs est segmentée ainsi :

- l'utilisateur dont l'UID est 0 est le root ;
- les utilisateurs dont l'UID est inférieur à une certaine valeur sont des utilisateurs systèmes
- les utilisateurs dont l'UID est compris entre cette valeur et 65533 sont des utilisateurs réels, cette plage peut quelquefois être elle-même subdivisée
- l'utilisateur dont l'UID est 65534 est « nobody » ou « nfsnobody » (pour les systèmes qui supportent les UID sur 32 bits, afin de conserver une compatibilité avec les anciens serveurs NFS)

❖ Souvent, à chaque utilisateur correspond un groupe :

- dont le GID est identique à l'UID de l'utilisateur ;
- dont le nom est le même que celui de l'utilisateur
- qui est le groupe primaire de l'utilisateur.

Le support des utilisateurs codés sur 32 bits commence à arriver dans les distributions récentes afin de porter le nombre d'utilisateurs possibles de 65535 à plus de 4 millions.

Ajout et suppression des groupes

Pour résumer …

❖ Créer un utilisateur consiste à réaliser plusieurs étapes :

- créer l'utilisateur et éventuellement son groupe primaire dans les fichiers passwd, shadow et group
- créer le répertoire principal de cet utilisateur (dans /home) et lui affecter les bons droits d'accès
- associer des quotas à l'utilisateur si nécessaire
- copier un certain nombre de fichiers standards depuis le répertoire /etc/skel vers le répertoire principal de l'utilisateur et leur affecter les bonnes permissions

❖ Ces opérations peuvent être réalisées :

- « à la main » (avec vi, mkdir, cp, chown, chfn, chsh, etc.), peu recommandé
- en ligne de commande avec principalement l'outil adduser
- à l'aide d'outils graphiques tels que Webmin ou le gestionnaires d'utilisateurs de RedHat

❖ Le principe reste le même pour la modification et la suppression des utilisateurs ou la gestion des groupes d'utilisateurs.

Plan

I. Introduction
II. Fichiers
III. Ajout et suppression des comptes utilisateurs
IV. Ajout et Suppression des groupes
V. Gestion des propriétés et des droits d'accès aux fichiers
VI. Répertoires et fichiers spéciaux

Gestion des propriétés et des droits d'accès aux fichiers

- Système multiutilisateur → savoir à qui appartient chaque fichier et si un utilisateur est autorisé ou non à effectuer des actions sur ce fichier.
- Cela se fait grâce à un système de permissions à trois niveaux : chaque fichier sur le disque appartient à un utilisateur et à un groupe d'utilisateurs et possède trois ensembles de permissions : une pour son propriétaire, une pour le groupe qui possède le fichier et une pour tous les autres
- ls → obtenir une liste du contenu de n'importe quel répertoire
- ls –l → beaucoup plus d'informations disponibles pour chaque fichier, y compris le type, la taille, la propriété et plus encore.

cap7

Gestion des propriétés et des droits d'accès aux fichiers

Chaque colonne de la sortie ci-dessus a une signification :

1. La *première* colonne de la liste indique le type de fichier et les permissions.

Par exemple, pour drwxrwxr-x:

Le premier caractère, d, indique le type de fichier.
Les trois caractères suivants, rwx, indiquent les permissions pour le propriétaire du fichier, également appelé *user* ou u.
Les trois caractères suivants, rwx, indiquent les permissions du *groupe* propriétaire du fichier, également appelé g.
Les trois derniers caractères, r-x, indiquent les permissions pour toute autre utilisateur, également appelée *others* ou o.

2. La *deuxième* colonne indique le nombre de liens physiques pointant vers ce fichier. Pour un répertoire, cela signifie le nombre de sous-répertoires, plus un lien vers lui-même (.) et le répertoire parent (..).
3. Les *troisième* et *quatrième* colonnes indiquent les informations relatives à la propriété : respectivement l'utilisateur et le groupe propriétaire du fichier.
5. La *cinquième* colonne indique la taille des fichiers, en octets.
6. La *sixième* colonne indique la date et l'heure précises, ou l'*horodatage*, de la dernière modification du fichier.
7. La *septième* et dernière colonne indique le nom du fichier.

Gestion des propriétés et des droits d'accès aux fichiers

- Si vous souhaitez voir les tailles de fichiers au format "lisible par l'homme", ajoutez le paramètre -h
capture ls –lh
- Pour n'afficher que les informations sur un ensemble spécifique de fichiers, ajoutez les noms de ces fichiers à ls :

$ **ls -lh HugeFile.zip test.sh capture**

- Si vous essayez d'obtenir des informations sur un répertoire en utilisant ls -l, vous obtiendrez à la place la liste du contenu du répertoire :

$ **ls -l Another_Directory/**

- Pour éviter cela et avoir les informations sur le répertoire lui-même, ajoutez le paramètre -d à ls :

$ **ls -l -d Another_Directory/**

- Pour voir les répertoires cachés:

$ **ls -l –a**

- Les répertoires **.** et **..** sont cependant spéciaux. **.** est un pointeur vers le répertoire courant, tandis que **..** est un pointeur vers le répertoire parent (le répertoire qui contient le répertoire courant). Sous Linux, chaque répertoire contient au moins ces deux répertoires spéciaux.

Gestion des propriétés et des droits d'accès aux fichiers

Types de fichiers

ls -l décrit le type de fichier.

Les trois types de fichiers les plus courants sont les suivants :

- **-** (fichier normal): Un fichier peut contenir des données de toute nature. Les fichiers peuvent être modifiés, déplacés, copiés et supprimés
- **d** (répertoire): Un répertoire contient d'autres fichiers ou répertoires et aide à organiser le système de fichiers. Techniquement, les répertoires sont un type de fichier particulier
- **l** (lien souple): Ce "fichier" est un pointeur vers un autre fichier ou répertoire ailleurs dans le système de fichiers

Gestion des propriétés et des droits d'accès aux fichiers

Types de fichiers

En plus de ceux-ci, il existe trois autres types de fichiers que vous devriez au moins connaître, mais qui ne sont pas couverts par cette leçon :

- **b** (périphérique à blocs): Ce fichier représente un périphérique virtuel ou physique, généralement des disques ou d'autres types de périphérique de stockage. Par exemple, le premier disque dur du système peut être représenté par /dev/sda.

- **c** (périphérique à caractères): Ce fichier représente un périphérique virtuel ou physique. Les terminaux (comme le terminal principal sur /dev/ttyS0) et les ports série sont des exemples courants de périphériques à caractères.

- **s** (socket): Les sockets servent de "conduits" transmettant les informations entre deux programmes.

Ne modifiez aucune des permissions sur les périphériques à blocs, les périphériques à caractères ou les sockets, à moins que vous ne sachiez ce que vous faites. Cela pourrait empêcher votre système de fonctionner !

Gestion des propriétés et des droits d'accès aux fichiers

Permissions:

- Dans la sortie de **ls -l**, les permissions de fichier sont affichées juste après le type de fichier, sous la forme de trois groupes de trois caractères chacun, dans l'ordre r, w et x.

Permissions sur les fichiers

Permission	Explication
r	Signifie *read* (lecture) et a une valeur octale de 4.Cela signifie qu'il est permis d'ouvrir un fichier et d'en lire le contenu.
w	Signifie *write* (écriture) et a une valeur octale de 2. Cela induit la permission de modifier ou de supprimer un fichier.
x	Signifie *execute* et a une valeur octale de 1, ce qui signifie que le fichier peut être exécuté en tant qu'exécutable ou script.

Ainsi, par exemple, un fichier avec les permissions rw- peut être lu et écrit, mais ne peut pas être exécuté.

Gestion des propriétés et des droits d'accès aux fichiers

Permissions sur les répertoires:

Permission	Explication
r	Signifie *read* et a une valeur octale de 4, ce qui signifie qu'il est permis de lire le contenu du répertoire, comme les noms de fichiers. Mais cela n'implique *pas* la permission de lire les fichiers eux-mêmes.
w	Signifie *write* et a une valeur octale de 2. Il s'agit de la permission de créer ou de supprimer des fichiers dans un répertoire, ou de modifier leurs noms, permissions et propriétaires. Si un utilisateur a la permission d'écrire sur un répertoire, il peut changer les permissions de n'importe quel fichier dans le répertoire, même s'il n'a pas de permissions sur le fichier ou si le fichier appartient à un autre utilisateur.
x	Signifie *execute* et a une valeur octale de 1, Il s'agit d'une permission d'entrer dans un répertoire, mais pas de lister ses fichiers (pour cela, la permission r est nécessaire).

Gestion des propriétés et des droits d'accès aux fichiers

Exemple:

$ **ls -ld Another_Directory/**

d--xr-xr-x 2 sr01 sr01 4,0K Dec 20 18:46 Another_Directory

Imaginez aussi que dans ce répertoire vous avez un script shell appelé hello.sh avec les permissions suivantes :

-rwxr-xr-x 1 sr01 sr01 33 Dec 20 18:46 hello.sh

Si vous êtes l'utilisateur sr01 et que vous essayez de lister le contenu de Another_Directory, vous obtiendrez un message d'erreur, car votre utilisateur n'a pas les droits de lecture pour ce répertoire :

$ **ls -l Another_Directory/**

ls: cannot open directory 'Another_Directory/': Permission denied

Cependant, l'utilisateur sr01 *a* les droits d'exécution, ce qui signifie qu'il peut entrer dans le répertoire. Par conséquent, l'utilisateur sr01 peut accéder aux fichiers à l'intérieur du répertoire, à condition qu'il ait les permissions correctes pour le fichier en question. Dans cet exemple, l'utilisateur a les permissions complètes pour le script hello.sh, ce qui lui permet d'*exécuter* le script, même s'il *ne peut pas* lire le contenu du répertoire qui le contient. On a uniquement besoin du nom complet du fichier.

Gestion des propriétés et des droits d'accès aux fichiers

Modification des permissions:

- **Chmod** est utilisé pour la modification des permissions et prend au moins deux paramètres: le premier présente les permissions à modifier et le deuxième indique le fichier ou le répertoire où la modification sera faite.

--> les permissions de modification peuvent être décrites de deux manières différentes, ou "modes" (mode symbolique et mode numérique).

Mode Symbolique:

- Il offre un contrôle fin, permettant d'ajouter ou de retirer une permission unique sans en modifier d'autres sur l'ensemble.

$ **chmod ug+rw-x,o-rwx sr01.txt**

-rw-rw---- 1 admin admin 765 Dec 20 21:25 sr01txt

- **En mode symbolique,** le(s) premier(s) caractère(s) indique(nt) les permissions que vous allez modifier : celles de l'utilisateur (u), du groupe (g), des autres (o) et/ou des trois ensemble (a).
- Ensuite, vous devez dire à la commande ce qu'elle doit faire : vous pouvez accorder une permission (+), révoquer une permission (-) ou lui donner une valeur spécifique (=).
- Enfin, vous précisez quelle permission vous souhaitez affecter : lecture (r), écriture (w) ou exécution (x).

Gestion des propriétés et des droits d'accès aux fichiers

Mode Symbolique:

- Lorsqu'elle est exécutée sur un répertoire, chmod ne modifie que les permissions du répertoire.
- Lorsque vous souhaitez modifier les permissions pour "tous les fichiers à l'intérieur d'un répertoire et de ses sous-répertoires" , ajoutez le paramètre -R après le nom de la commande et avant les permissions à modifier.

Mode Numérique:

- Les permissions sont spécifiées d'une manière différente : sous la forme d'une valeur numérique à trois chiffres en notation octale, un système numérique en base 8.
- Chaque permission a une valeur correspondante, et elles sont spécifiées dans l'ordre suivant : d'abord vient lecture (r), qui est 4, puis écriture (w), qui est 2 et enfin exécution (x), représentée par 1. S'il n'y a pas de permission, utilisez la valeur zéro (0). Ainsi, une permission rwx serait 7 (4+2+1) et r-x serait 5 (4+0+1).
- Le premier des trois chiffres de l'ensemble des permissions représente les permissions pour l'utilisateur (u), le deuxième pour le groupe (g) et le troisième pour les autres (o).

```
$ chmod 660 text.txt
$ ls -l text.txt
-rw-rw---- 1 sr01 sr01 765 Dec 20 21:25 sr01.txt
```

Gestion des propriétés et des droits d'accès aux fichiers

Modification des propriétés:

- La commande **chown** est utilisée pour modifier la propriété d'un fichier ou d'un répertoire.
- Syntaxe: **chown utilisateur:groupe fichiercible**

$ **ls -l text.txt**

-rw-rw---- 1 sr01 sr01 1881 Dec 10 15:57 sr01.txt

- L'utilisateur qui possède le fichier est sr01, et le groupe est également sr01. Maintenant, modifions le groupe propriétaire du fichier pour un autre groupe, comme students :

$ **chown admin:students text.txt**

$ **ls -l text.txt**

-rw-rw---- 1 admin students 1881 Dec 10 15:57 text.txt

- **chgrp students text.txt** pour ne changer que le groupe

Gestion des propriétés et des droits d'accès aux fichiers

Interroger les groupes:

- Avant de changer la propriété d'un fichier, il peut être utile de savoir quels groupes existent sur le système, quels utilisateurs sont membres d'un groupe et à quels groupes appartient un utilisateur
- Pour voir quels groupes existent sur votre système, il suffit de taper groups :

  $ **groups**

  Root sr01 students cdrom sudo dip plugdev lpadmin sambashare
- Et si vous voulez savoir à quels groupes appartient un utilisateur, ajoutez le nom d'utilisateur comme paramètre :

  $ **groups sr01**

  sr01 : sr01 students cdrom sudo dip plugdev lpadmin sambashare
- Pour faire l'inverse, en affichant quels utilisateurs appartiennent à un groupe, utilisez **groupmems**, le paramètre -g spécifie le groupe, et -l liste tous ses membres :

  $ **sudo groupmems -g cdrom -l**

  sr01
- **groupmems** ne peut être exécuté qu'en tant que root **!**

Gestion des propriétés et des droits d'accès aux fichiers

Permissions spéciales:

- Outre les permissions de lecture, d'écriture et d'exécution pour les utilisateurs, les groupes et les autres, chaque fichier peut avoir trois autres *permissions spéciales* qui peuvent modifier le fonctionnement d'un répertoire ou l'exécution d'un programme (Sticky Bit, Set UID, Set GID)

Sticky Bit:

- Le sticky bit (bit collant), également appelé *étiquette de suppression restreinte*, a la valeur octale **1** et, en mode symbolique, est représenté par un **t** dans les permissions des autres

- Cela ne s'applique qu'aux répertoires et, sous Linux, cela empêche les utilisateurs de supprimer ou de renommer un fichier dans un répertoire, à moins qu'ils ne soient propriétaires de ce fichier ou de ce répertoire

- Les répertoires avec le sticky bit activé montrent un t remplaçant le x sur les permissions des *autres* sur la sortie de ls –l
- En mode numérique, les permissions spéciales sont spécifiées à l'aide d'une "notation à 4 chiffres", le premier chiffre représentant la permission spéciale sur laquelle agir. Par exemple, pour définir le sticky bit (valeur 1) pour le répertoire Another_Directory en mode numérique, avec les permissions 755, la commande serait :

$ **chmod 1755 Another_Directory**

Gestion des propriétés et des droits d'accès aux fichiers

Sticky Bit:

- En mode numérique, les permissions spéciales sont spécifiées à l'aide d'une "notation à 4 chiffres", le premier chiffre représentant la permission spéciale sur laquelle agir. Par exemple, pour définir le sticky bit (valeur 1) pour le répertoire Another_Directory en mode numérique, avec les permissions 755, la commande serait :

$ **chmod 1755 Another_Directory**
$ **ls -ld Another_Directory**
drwxr-xr-t 2 sr01 sr01 4,0K Dec 10 18:46 Another_Directory

Gestion des propriétés et des droits d'accès aux fichiers

Set GID:

- Le Set GID, également connu sous le nom SGID ou bit Set Group ID, a la valeur octale 2 et en mode symbolique est représenté par un **s** sur les permissions du *groupe*
- Cela peut s'appliquer aux fichiers ou répertoires exécutables
- Sur les fichiers exécutables, il accordera au processus résultant de l'exécution du fichier l'accès aux privilèges du groupe qui possède le fichier
- Lorsqu'il est appliqué aux répertoires, il fera en sorte que chaque fichier ou répertoire créé sous lui hérite du groupe du répertoire parent
- Les fichiers et répertoires avec un bit SGID montrent un s remplaçant le x sur les permissions du *groupe* sur la sortie de ls –l
- Pour ajouter des permissions SGID à un fichier en mode symbolique, la commande serait:

```
$ chmod g+s test.sh
$ ls -l test.sh
-rwxr-sr-x 1 admin root 33 Dec 11 10:36 test.sh
```

Gestion des propriétés et des droits d'accès aux fichiers

Set GID:

Exemple:

- Supposons que nous ayons un répertoire appelé Sample_Directory, appartenant à l'utilisateur admin et au groupe users, avec la structure de permission suivante :

  $ **ls -ldh Sample_Directory/**

  drwxr-xr-x 2 admin users 4,0K Dec 18 17:06 Sample_Directory/

- Maintenant, entrons dans ce répertoire, et à l'aide de la commande touch créons un fichier vide à l'intérieur. Le résultat serait :

  $ **cd Sample_Directory/**

  $ **touch newfile**

  $ **ls -lh newfile**

  -rw-r--r-- 1 admin admin 0 Dec18 17:11 newfile

- Comme on peut le voir, le fichier est la propriété de l'utilisateur admin et du groupe admin. Mais, si le répertoire avait les permissions SGID définies, le résultat serait différent.

Gestion des propriétés et des droits d'accès aux fichiers

Set GID:

Exemple:

- Tout d'abord, ajoutons le bit SGID au répertoire Sample_Directory et vérifions les résultats :

  $ **sudo chmod g+s Sample_Directory/**

  $ **ls -ldh Sample_Directory/**

  drwxr-sr-x 2 admin users 4,0K Dec 18 17:27 Sample_Directory/

- Le s sur les permissions de groupe indique que le bit SGID est activé. Maintenant, rentrons dans ce répertoire et, à nouveau, créons un fichier vide avec la commande touch :

  $ **cd Sample_Directory/**

  $ **touch emptyfile**

  $ **ls -lh emptyfile**

  -rw-r--r-- 1 admin users 0 Jan 18 17:31 emptyfile

- Comme on peut le voir, le groupe qui possède le fichier est users. Cela est dû au fait que le bit SGID a fait que le fichier hérite du groupe propriétaire de son répertoire parent, qui est users.

Gestion des propriétés et des droits d'accès aux fichiers

Set UID:

- Le SUID, également connu sous le nom de Set User ID, a la valeur octale 4 et est représenté par un s sur les permissions de l'*utilisateur* en mode symbolique.
- Il s'applique uniquement aux fichiers et son comportement est similaire au bit SGID, mais le processus s'exécutera avec les privilèges de l'utilisateur qui possède le fichier.
- Les fichiers avec le bit SUID montrent un s remplaçant le x sur les permissions de l'utilisateur sur la sortie de ls -l :

  $ **ls -ld test.sh**

  -rwsr-xr-x 1 admin admin 33 Dec 11 10:36 test.sh

- Vous pouvez combiner plusieurs permissions spéciales dans un même paramètre en les additionnant. Ainsi, pour mettre SGID (valeur 2) et SUID (valeur 4) en mode numérique pour le script test.sh avec les permissions 755, vous devez taper :

  $ **chmod 6755 test.sh**

  $ **ls -lh test.sh**

  -rwsr-sr-x 1 admin admin 66 Dec 18 17:29 test.sh

Plan

I. Introduction
II. Fichiers
III. Ajout et suppression des comptes utilisateurs
IV. Ajout et Suppression des groupes
V. Gestion des propriétés et des droits d'accès aux fichiers
VI. Répertoires et fichiers spéciaux

Répertoires et fichiers spéciaux

- Certains fichiers reçoivent un traitement spécial, soit en raison de l'endroit où ils sont stockés, comme les fichiers temporaires, soit en raison de la manière dont ils interagissent avec le système de fichiers, comme les liens

Fichiers Temporaires

Les fichiers temporaires sont des fichiers utilisés par les programmes pour stocker des données qui ne sont nécessaires que pour une courte durée. Par exemple les données de processus en cours d'exécution, de journaux de crash, de fichiers de travail d'une sauvegarde automatique, de fichiers intermédiaires utilisés lors d'une conversion de fichier, de fichiers de cache, etc.

Localisation des fichiers temporaires:

La version 3.0 du *FHS* (Filesystem Hierarchy Standard) définit des emplacements standard pour les fichiers temporaires sur les systèmes Linux. Il est recommandé de suivre les conventions établies par le FHS en cas d'écriture de données temporaires sur le disque.

/tmp: Selon le FHS, les programmes ne doivent pas supposer que les fichiers écrits ici seront conservés entre les exécutions d'un programme. La recommandation est que ce répertoire soit effacé (tous les fichiers sont effacés) lors du démarrage du système, bien que cela *ne soit pas obligatoire*.

Répertoires et fichiers spéciaux

/var.tmp: Un autre emplacement pour les fichiers temporaires, mais celui-ci *ne doit pas être effacé* pendant le démarrage du système, c'est-à-dire que les fichiers stockés ici persisteront généralement entre les redémarrages.

/run: Ce répertoire contient les données des variables d'exécution utilisées par les processus en cours, telles que les fichiers d'identification des processus (.pid). Les programmes qui ont besoin de plus d'un fichier d'exécution peuvent créer des sous-répertoires ici. Cet emplacement *doit être effacé* lors du démarrage du système. Le but de ce répertoire était autrefois servi par /var/run, et sur certains systèmes, /var/run peut être un lien symbolique vers /run.

Permissions sur les fichiers temporaires:

- Le fait d'avoir des répertoires temporaires à l'échelle du système sur un système multi-utilisateurs présente quelques difficultés concernant les permissions d'accès.
- Comment pourrions-nous empêcher un utilisateur d'effacer ou de modifier des fichiers créés par un autre ?
- La solution est une permission spéciale appelée *sticky bit* (bit collant), qui s'applique à la fois aux répertoires et aux fichiers.

Répertoires et fichiers spéciaux

Les liens

- Dans linux, tout est traité comme un fichier. Mais il existe un type de fichier *spécial*, appelé *lien*, et il y a deux types de liens sur un système Linux : lien symbolique et lien solide.
- **Liens symboliques**, appelés aussi *liens souples*, ils pointent vers le chemin d'un autre fichier. Si vous supprimez le fichier vers lequel le lien pointe (appelé *cible*), le lien existera toujours, mais il "cessera de fonctionner", car il pointe maintenant vers "rien".

- **Liens solides**, c'est comme un deuxième nom pour le fichier original. Ce *ne* sont *pas* des doublons, mais plutôt une entrée supplémentaire dans le système de fichiers pointant vers le même endroit (inode) sur le disque.

- Ps: Un *inode* est une structure de données qui stocke les attributs d'un objet (comme un fichier ou un répertoire) sur un système de fichiers. Parmi ces attributs figurent le nom du fichier, les permissions, la propriété et les blocs du disque sur lesquels les données de l'objet sont stockées. Considérez cela comme une entrée dans un index, d'où le nom, qui vient de "index node".

Répertoires et fichiers spéciaux

Liens solides:

- La commande pour créer un lien solide sur Linux est ln. La syntaxe de base est :

  $ **ln CIBLE NOM_DU_LIEN**

- La CIBLE doit déjà exister (c'est le fichier vers lequel le lien pointera), et si la cible ne se trouve pas dans le répertoire actuel, ou si vous voulez créer le lien ailleurs, vous devez indiquer le chemin complet vers celui-ci. Par exemple, la commande :

  $ **ln target.txt /home/sr01/Documents/hardlink**

- Cette commande va créer un fichier nommé hardlink dans le répertoire /home/sr01/Documents/, lié au fichier target.txt du répertoire courant

- Les liens solides sont des entrées dans le système de fichiers qui ont des noms différents mais qui pointent vers les mêmes données sur le disque.
- Si vous modifiez le contenu de l'un des noms, le contenu de tous les autres noms pointant vers ce fichier change puisque tous ces noms pointent vers les mêmes données. Si vous supprimez l'un des noms, les autres noms continueront à fonctionner. Pourquoi ??

Répertoires et fichiers spéciaux

Liens solides:

- Cela se produit parce que lorsque vous "supprimez" un fichier, les données ne sont pas réellement effacées du disque. Le système supprime simplement l'entrée de la table du système de fichiers qui pointe sur l'inode correspondant aux données du disque. Mais si vous avez une deuxième entrée pointant vers le même inode, vous pouvez toujours accéder aux données.

- Vous pouvez vérifier cela en utilisant le paramètre -i de ls. Considérez le contenu suivant d'un répertoire :

$ **ls -li** total 224

3806696 -r--r--r-- 2 sr01 sr01 111702 Dec 7 10:13 hardlink

3806696 -r--r--r-- 2 sr01 sr01 111702 Dec 7 10:13 target.txt

- Le numéro précédant les permissions est le numéro d'inode. Vous voyez que le fichier hardlink et le fichier target.txt ont le même numéro (3806696) ? C'est parce que l'un est un lien solide de l'autre.
- Chaque lien solide pointant vers un fichier augmente le *nombre de liens* du fichier. C'est le nombre juste après les permissions sur la sortie de ls -l. Par défaut, chaque fichier a un nombre de liens de 1 (les répertoires ont un nombre de 2).

Répertoires et fichiers spéciaux

Liens solides:

- Contrairement aux liens symboliques, vous ne pouvez créer que des liens solides vers des fichiers, et le lien et la cible doivent tous deux résider dans le même système de fichiers !

- Comme les liens solides sont traités comme des fichiers ordinaires, ils peuvent être supprimés avec rm et renommés ou déplacés dans le système de fichiers avec mv. Et puisqu'un lien solide pointe vers le même inode de la cible, il peut être déplacé librement, sans crainte de "casser" le lien.

Répertoires et fichiers spéciaux

Liens symboliques:

- Contrairement aux liens symboliques, vous ne pouvez créer que des liens solides vers des fichiers, et le lien et la commande utilisée pour créer un lien symbolique est également ln, mais avec le paramètre -s ajouté. Ainsi :

  $ **ln -s target.txt /home/sr01/Documents/softlink**

- Cela créera un fichier nommé softlink dans le répertoire /home/sr01/Documents/, pointant vers le fichier target.txt dans le répertoire courant.
- Comme pour les liens solides, vous pouvez omettre le nom du lien pour créer un lien portant le même nom que la cible dans le répertoire courant.
- Les liens symboliques pointent vers un autre chemin dans le système de fichiers. Vous pouvez créer des liens symboliques vers des fichiers et des répertoires, même sur des partitions différentes.

  $ **ls -lh**
  total 112K
  -rw-r--r-- 1 sr01 sr01 110K Dec 7 10:13 target.txt
  l*rwxrwxrwx 1 sr01 sr01 12 Dec 7 10:14* ***softlink -> target.txt***

- Le premier caractère sur les permissions du fichier softlink est l, indiquant un lien symbolique. Et juste après le nom du fichier, nous voyons le nom de la cible vers laquelle le lien pointe « le fichier target.txt ».

Répertoires et fichiers spéciaux

Liens symboliques:

- Les liens symboliques peuvent être supprimés à l'aide de rm et déplacés ou renommés à l'aide de mv.
- Cependant, il faut faire particulièrement attention lors de leur création, pour éviter de « casser » le lien s'il est déplacé de son emplacement d'origine
- Supposons que nous ayons un fichier nommé original.txt dans le répertoire courant, et que nous voulions créer un lien symbolique vers celui-ci appelé softlink. Nous pourrions utiliser :

  $ **ln -s original.txt softlink**
- Résultat:

  $ **ls -lh** total 112K
  -r--r--r-- 1 sr01 sr01 110K Dec 7 10:13 original.txt
  lrwxrwxrwx 1 sr01 sr01 12 Jun 7 19:23 softlink -> original.txt
- Mais, si nous déplaçons le lien vers le répertoire parent et essayons d'afficher son contenu en utilisant la commande less :

  $ **mv softlink ../**
  $ **less ../softlink**
  ../softlink: No such file or directory

Répertoires et fichiers spéciaux

Liens symboliques:

- Comme le chemin d'accès au fichier original.txt n'a pas été spécifié, le système suppose qu'il se trouve dans le même répertoire que le lien. Lorsque cela n'est plus vrai, le lien cesse de fonctionner.

- Pour éviter cela, il faut toujours spécifier le chemin complet vers la cible lors de la création du lien :

$ **ln -s /home/sr01/Documents/original.txt softlink**