

La Gestion des droits des fichiers et répertoires

Sous Ubuntu, la gestion des droits des fichiers et répertoires est cruciale pour la sécurité et le bon fonctionnement du système. Voici un aperçu des permissions, de leur configuration et des commandes courantes pour les manipuler.

1. Structure des Permissions

Sous Ubuntu, chaque fichier et répertoire a des permissions associées, définies pour trois catégories d'utilisateurs :

- **Propriétaire** (user ou `u`) : la personne ayant créé le fichier/répertoire.
- **Groupe** (group ou `g`) : ensemble d'utilisateurs autorisés.
- **Autres** (other ou `o`) : tous les autres utilisateurs du système.

Les permissions sont représentées par trois types d'autorisations :

- **Lecture** (`r` pour read) : capacité de lire le contenu d'un fichier ou de lister un répertoire.
- **Écriture** (`w` pour write) : autorisation de modifier un fichier ou de créer des éléments dans un répertoire.
- **Exécution** (`x` pour execute) : permission d'exécuter un fichier (utile pour les scripts et les binaires) ou d'accéder aux sous-répertoires.

Exemple d'affichage des permissions via la commande `ls -l` :

```
-rwxr-xr-- 1 utilisateur groupe taille date fichier
```

- Le premier caractère indique le type (fichier `-`, répertoire `d`, etc.)
- Les 9 caractères suivants montrent les permissions pour le propriétaire, le groupe et les autres (`rwx`, `r-x`, `r--` ici).

2. Gestion des Permissions avec `chmod`

La commande `chmod` (change mode) permet de modifier les droits d'accès. Deux méthodes sont courantes :

- **Symbole** (symbolic) : utilise les lettres pour ajouter ou retirer des permissions.
- **Octal** (numérique) : chaque droit est représenté par un chiffre.

Utilisation Symbolique

Les opérateurs sont `+` pour ajouter, `-` pour retirer, et `=` pour définir spécifiquement des permissions. Exemple :

```
chmod u+x fichier # Ajoute le droit d'exécution pour le propriétaire
chmod g-w fichier # Retire le droit d'écriture pour le groupe
chmod o=r fichier # Donne uniquement le droit de lecture aux autres
```

Utilisation Octale

L'octal associe chaque droit (lecture, écriture, exécution) à un chiffre : `4` (lecture), `2` (écriture), `1` (exécution).

- `7` (`rwx`) pour tous les droits.
- `5` (`r-x`) pour lecture et exécution.
- `0` (`---`) pour aucun droit.

Exemples :

```
chmod 755 fichier # Propriétaire : tous les droits, groupe et autres : lecture et exécution
chmod 644 fichier # Propriétaire : lecture et écriture, groupe et autres : lecture uniquement
```

3. Changer le Propriétaire et le Groupe avec `chown` et `chgrp`

Pour changer le propriétaire d'un fichier ou d'un répertoire :

```
chown nouveau_proprietaire fichier
```

Pour changer le groupe associé :

```
chgrp nouveau_groupe fichier
```

Combiner les deux dans `chown` :

```
chown nouveau_proprietaire:nouveau_groupe fichier
```

4. Permissions Avancées : `sudo`, ACL, et `umask`

- **Sudo** : Les utilisateurs non root peuvent utiliser `sudo` pour exécuter des commandes avec des privilèges élevés, par exemple, pour modifier des fichiers système.
- **ACL (Access Control List)** : En plus des permissions de base, ACL permet d'accorder des permissions personnalisées pour des utilisateurs spécifiques :

```
setfacl -m u:utilisateur:rwx fichier # Ajoute des permissions spécifiques  
getfacl fichier # Vérifie les permissions ACL
```

- **Umask** : Définit les permissions par défaut pour les nouveaux fichiers et répertoires. Le `umask` est soustrait des permissions maximales (777 pour répertoires et 666 pour fichiers).

```
umask 022 # Par défaut, enlève l'écriture pour le groupe et les autres
```

🔄 Révision #2

★ Créé 26 octobre 2024 07:26:00 par Marc Leroi

✎ Mis à jour 26 octobre 2024 07:27:24 par Marc Leroi