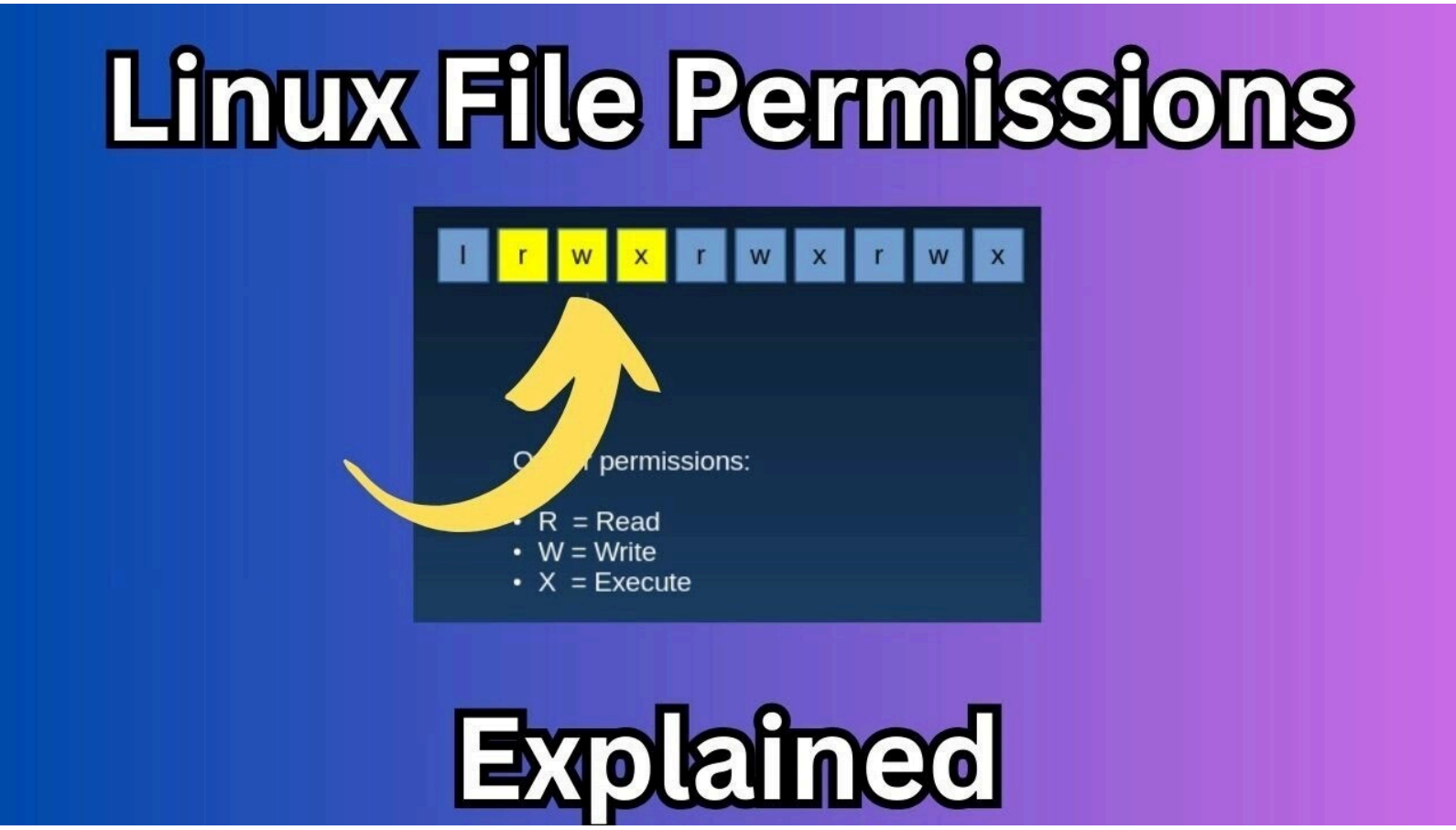
Lucas Vidal Genoud

BTSSIO

Les permission Linux



Sommaire :

I. Introduction aux permissions

Dans les systèmes Linux, chaque fichier et dossier possède des droits d'accès qui définissent **qui peut lire, écrire ou exécuter** ce fichier. Ces droits s'appliquent à trois types d'utilisateurs :

- Le **propriétaire** (user) : généralement la personne ayant créé le fichier.
- Le **groupe** (group) : un groupe d'utilisateurs auquel le fichier peut être attribué.
- Les **autres** (others) : tous les autres utilisateurs du système.

Il existe trois types d'autorisations :

- **Lecture** (read), notée par la lettre **r** : permet d'ouvrir un fichier ou de lister le contenu d'un dossier.
- **Écriture** (write), notée par la lettre **w** : permet de modifier le contenu d'un fichier ou de créer/supprimer des fichiers dans un dossier.
- **Exécution** (execute), notée par la lettre **x** : permet d'exécuter un fichier (comme un script ou un programme) ou d'entrer dans un dossier.

II. Affichage des permissions

Lorsque l'on affiche les fichiers avec détails, on voit une série de dix caractères, comme par exemple : `-rwxr-xr--`. Cela se lit ainsi :

- Le premier caractère indique le type de fichier : un tiret pour un fichier normal, une lettre `d` pour un dossier.
- Les trois caractères suivants indiquent les droits du **propriétaire**.
- Les trois caractères du milieu sont ceux du **groupe**.
- Les trois derniers concernent les **autres**.

Dans l'exemple `-rwxr-xr--` :

- Le propriétaire a tous les droits (lecture, écriture, exécution).
- Le groupe peut lire et exécuter, mais pas écrire.
- Les autres peuvent seulement lire.

III. Notation octale des droits

Les autorisations peuvent également être exprimées en notation **octale**. Chaque type de droit est représenté par une valeur numérique :

- Lecture (r) vaut 4
- Écriture (w) vaut 2
- Exécution (x) vaut 1

On additionne ces valeurs pour obtenir la permission d'un groupe :

- `7` signifie lecture, écriture et exécution (4+2+1)
- `6` signifie lecture et écriture (4+2)
- `5` signifie lecture et exécution (4+1)
- `4` signifie lecture seule
- `0` signifie aucun droit

Un ensemble d'autorisations est donc noté sur trois chiffres, par exemple `754` :

- Le premier chiffre est pour le propriétaire
- Le deuxième pour le groupe
- Le troisième pour les autres

Dans 754, cela signifie :

- Le propriétaire a les droits complets (rwx)
- Le groupe peut lire et exécuter (r-x)
- Les autres peuvent seulement lire (r--)

IV. Modification des droits

Les autorisations peuvent être modifiées soit en notation **symbolique**, soit en notation **octale**.

En symbolique, on utilise les lettres u pour user (propriétaire), g pour group, o pour others, et a pour all (tous). On peut ajouter, retirer ou définir des droits avec les signes +, - ou =.

Par exemple :

- Ajouter l'exécution au propriétaire
- Retirer l'écriture au groupe
- Donner uniquement la lecture aux autres

En notation octale, il suffit d'indiquer la valeur numérique correspondant aux droits souhaités.

V. Propriétaire et groupe

Chaque fichier ou dossier a un **propriétaire** et un **groupe** associés.

On peut modifier :

- Le **propriétaire** seul
- Le **groupe** seul
- Ou les deux en même temps

- Cela permet de gérer les accès dans des environnements multi-utilisateurs.

VI. Signification des droits sur les dossiers

Les droits n'ont pas exactement la même signification pour les dossiers que pour les fichiers :

- Le droit de **lecture** permet de voir la liste des fichiers présents dans le dossier.
- Le droit d'**écriture** permet de créer, renommer ou supprimer des fichiers dans le dossier.
- Le droit d'**exécution** permet de « traverser » le dossier, c'est-à-dire de l'ouvrir ou d'y accéder via le terminal.

Ainsi, pour pouvoir entrer dans un dossier, il faut impérativement avoir le droit d'exécution sur ce dossier.

VII. Permissions spéciales

Il existe trois types de permissions spéciales sous Linux :

1. Setuid

Appliqué à un **fichier exécutable**, cela signifie que lorsqu'un utilisateur l'exécute, le processus est lancé avec les **droits du propriétaire du fichier**, et non ceux de l'utilisateur courant. Cela est utile pour certains programmes.

2. Setgid

Appliqué à un **fichier exécutable**, cela donne les **droits du groupe propriétaire** au moment de l'exécution.

Appliqué à un **dossier**, cela signifie que tous les fichiers créés à l'intérieur auront le **même groupe** que le dossier, ce qui est très utile dans les environnements collaboratifs.

3. Sticky bit

Cette permission s'applique uniquement aux **dossiers**. Elle permet à **chaque utilisateur de supprimer uniquement les fichiers qu'il a lui-même créés**, même s'il a les droits d'écriture sur le dossier. C'est une protection utile sur des dossiers partagés comme `/tmp`.

VIII. Application récursive

Il est souvent nécessaire d'appliquer les permissions à un dossier et à **tout son contenu** (sous-dossiers et fichiers inclus). Cela se fait de manière récursive et permet de garantir des droits homogènes dans un arborescence.

IX. Cas pratiques

Voici quelques exemples typiques d'utilisation des permissions :

- Pour rendre un script exécutable uniquement par son propriétaire, on attribue les droits complets au propriétaire, et aucun aux autres.
- Pour créer un dossier accessible à un groupe de travail mais pas aux autres utilisateurs, on attribue les droits complets au propriétaire et au groupe, et aucun aux autres.
- Pour un fichier public en lecture seule, on donne la lecture à tous mais l'écriture uniquement au propriétaire.

X. Utilisateurs et groupes

Un utilisateur appartient souvent à plusieurs **groupes**. Ces groupes déterminent les accès qu'il peut avoir aux ressources partagées. Le système Linux permet d'organiser les autorisations efficacement en jouant à la fois sur les propriétaires, les groupes, et les autres.