Permissions de fichiers sous Linux

Description du projet

Au sein de mon organisation, l'équipe de recherche manipule des données sensibles stockées dans le répertoire `project`. On s'est rendu compte que les permissions actuelles étaient un peu "portes ouvertes" : elles ne correspondaient plus du tout aux besoins de sécurité. En gros, trop de gens pouvaient voir ou modifier des fichiers qui ne les concernaient pas.

Vérifier les détails des fichiers et des répertoires

Avant de tout chambouler, j'ai dû regarder précisément comment les accès étaient configurés. Pour ça, je me suis placé dans le répertoire de l'équipe et j'ai utilisé une commande de base, mais ultra-puissante : `ls -la`.

```
researcher2@5d738f0f927b:~/projects$ ls -la
total 32
drwxr-xr-x 3 researcher2 research_team 4096 Dec  2 15:27 .
drwxr-xr-x 3 researcher2 research_team 4096 Dec  2 15:27 ..
-rw--w---- 1 researcher2 research_team   46 Dec  2 15:27 .project_x.txt
drwx--x--- 2 researcher2 research_team 4096 Dec  2 15:27 drafts
-rw-rw-rw- 1 researcher2 research_team   46 Dec  2 15:27 project_k.txt
-rw-r----- 1 researcher2 research_team   46 Dec  2 15:27 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 Dec  2 15:27 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 Dec  2 15:27 project_t.txt
researcher2@5d738f0f927b:~/projects$
```

Ce que j'ai découvert :

En analysant le résultat (voir ma capture d'écran), j'ai tout de suite repéré plusieurs éléments importants pour ma mission :

1. Un dossier nommé `drafts` (brouillons).
2. Un fichier caché stratégique : `.project_x.txt`.
3. Cinq autres fichiers de projet classiques.

En jetant un œil à la première colonne, j'ai pu identifier immédiatement les fichiers dont les permissions étaient trop larges. Par exemple, on voit bien que certains fichiers sont en "lecture/écriture" pour tout le monde, ce qui est exactement ce que je dois corriger pour sécuriser le système.

Décrivez la chaîne d'autorisations

Pour chaque élément, le système affiche une série de 10 caractères que j'ai pris le temps de décortiquer pour identifier les vulnérabilités :

- **Le type de fichier (1er caractère) :** Un `d` indique un répertoire (comme `drafts`), tandis qu'un trait d'union `-` signifie qu'il s'agit d'un fichier classique.
- **L'utilisateur / Propriétaire (caractères 2 à 4) :** C'est ici que je vois ce que le créateur du fichier peut faire.
- **Le groupe (caractères 5 à 7) :** C'est une zone critique. Si le groupe de recherche a trop de droits, n'importe quel membre peut modifier un travail sensible par erreur.
- **Les "autres" (caractères 8 à 10) :** C'est le niveau le plus dangereux. Si on voit des `r` ou des `w` ici, cela signifie que n'importe qui sur le système peut lire ou modifier le fichier.

Un exemple concret d'analyse : Prenons le fichier `project_t.txt` que j'ai audité. Sa chaîne est `-rw-rw-r--`.

- C'est un fichier ordinaire (`-`).
- L'utilisateur et le groupe peuvent le lire et le modifier (`rw`).
- Par contre, tous les autres utilisateurs du système ne peuvent que le lire (`r--`), sans pouvoir y toucher.

Cette étape de diagnostic était essentielle : elle m'a permis de repérer exactement où la sécurité faisait défaut avant de sortir les "outils de réparation" comme la commande `chmod`.

Modifier les permissions du fichier

Après avoir fait mon état des lieux, il était temps de passer aux choses sérieuses : sécuriser les fichiers. La règle de l'organisation était claire : **personne d'autre que le propriétaire et le groupe ne doit pouvoir modifier les fichiers de recherche.**

Le problème identifié : En regardant mes notes précédentes, j'ai vu que le fichier `project_k.txt` était beaucoup trop exposé. Sa chaîne de permissions indiquait que n'importe quel utilisateur du système pouvait écrire dedans. C'est une porte ouverte aux erreurs ou aux modifications malveillantes.

```
researcher2@5d738f0f927b:~/projects$ chmod o-w project_k.txt
researcher2@5d738f0f927b:~/projects$ ls -la
total 32
drwxr-xr-x 3 researcher2 research_team 4096 Dec  2 15:27 .
drwxr-xr-x 3 researcher2 research_team 4096 Dec  2 15:27 ..
-rw--w---- 1 researcher2 research_team   46 Dec  2 15:27 .project_x.txt
drwx--x--- 2 researcher2 research_team 4096 Dec  2 15:27 drafts
-rw-rw-r-- 1 researcher2 research_team   46 Dec  2 15:27 project_k.txt
-rw-r----- 1 researcher2 research_team   46 Dec  2 15:27 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 Dec  2 15:27 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 Dec  2 15:27 project_t.txt
researcher2@5d738f0f927b:~/projects$
```

Ma démarche pour sécuriser le fichier :

1. **L'exécution de la commande :** J'ai utilisé `chmod o-w project_k.txt`.
    - Le **o** cible les "others" (les autres utilisateurs).
    - Le **-w** leur retire explicitement le droit d'écriture (*write*).
2. **La preuve par l'image :** Comme on le voit sur ma capture d'écran, j'ai immédiatement enchaîné avec un `ls -la` pour confirmer le changement.
3. **Le résultat :** La chaîne de caractères pour `project_k.txt` est passée de `-rw-rw-rw-` à `-rw-rw-r--`.

Ce que cela change concrètement : Désormais, le fichier est verrouillé. Les membres de l'équipe peuvent toujours lire les données pour collaborer, mais le risque qu'un utilisateur externe vienne modifier ou effacer le contenu de `project_k.txt` est maintenant éliminé. C'est une petite commande, mais c'est elle qui garantit l'intégrité de notre travail de recherche.

Modifier les permissions d'un fichier caché

Pour cette étape, j'ai dû m'occuper d'un cas un peu spécial : le fichier `.project_x.txt`. En Linux, le point devant le nom signifie que c'est un fichier caché. L'équipe de recherche l'a archivé et ne voulait plus que quiconque puisse le modifier, mais il fallait que l'utilisateur et le groupe puissent encore consulter les données.

Le défi technique : Modifier les permissions d'un fichier caché demande la même précision, mais il ne faut surtout pas oublier le point dans la commande, sinon le système ne trouvera rien.

```
researcher2@3213bbc1d047:~/projects$ chmod u-w,g-w,g+r .project_x.txt
researcher2@3213bbc1d047:~/projects$ ls -la
total 32
drwxr-xr-x 3 researcher2 research_team 4096 Dec 20 15:36 .
drwxr-xr-x 3 researcher2 research_team 4096 Dec 20 15:36 ..
-r--r----- 1 researcher2 research_team   46 Dec 20 15:36 .project_x.txt
drwx--x--- 2 researcher2 research_team 4096 Dec 20 15:36 drafts
-rw-rw-rw- 1 researcher2 research_team   46 Dec 20 15:36 project_k.txt
-rw-r----- 1 researcher2 research_team   46 Dec 20 15:36 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 Dec 20 15:36 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 Dec 20 15:36 project_t.txt
researcher2@3213bbc1d047:~/projects$
```

Mon intervention chirurgicale avec `chmod` :

1. **La commande combinée :** J'ai saisi `chmod u-w,g-w,g+r .project_x.txt`.
    - **`u-w` et `g-w`** : J'ai retiré le droit d'écriture à l'utilisateur propriétaire et au groupe. Fini les modifications accidentelles !
    - **`g+r`** : Je me suis assuré que le groupe gardait bien son accès en lecture pour pouvoir consulter l'archive.
2. **La vérification immédiate :** Comme toujours, j'ai relancé un `ls -la`. Sans le `-a`, je n'aurais pas pu voir si mes changements sur ce fichier caché avaient fonctionné.
3. **Le constat :** Le fichier est passé de `-rw--w----` à `-r--r-----`. Mission accomplie : l'archive est désormais protégée en "lecture seule" pour les bonnes personnes.

Modifier les permissions du répertoire

Pour terminer ce nettoyage, je me suis attaqué au répertoire `drafts`. La consigne de sécurité était stricte : seul le propriétaire (`researcher2`) doit avoir le droit d'exécuter des fichiers ou de parcourir ce dossier. Personne d'autre, pas même les membres du groupe, ne doit avoir ces privilèges.

```
researcher2@5d738f0f927b:~/projects$ chmod g-x drafts
researcher2@5d738f0f927b:~/projects$ ls -la
total 32
drwxr-xr-x 3 researcher2 research_team 4096 Dec  2 15:27 .
drwxr-xr-x 3 researcher2 research_team 4096 Dec  2 15:27 ..
-r--r----- 1 researcher2 research_team   46 Dec  2 15:27 .project_x.txt
drwx------ 2 researcher2 research_team 4096 Dec  2 15:27 drafts
-rw-rw-r-- 1 researcher2 research_team   46 Dec  2 15:27 project_k.txt
-rw-r----- 1 researcher2 research_team   46 Dec  2 15:27 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 Dec  2 15:27 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 Dec  2 15:27 project_t.txt
researcher2@5d738f0f927b:~/projects$
```

Mon intervention finale :

1. **Le retrait des droits :** J'ai utilisé la commande `chmod g-x drafts`.
    - En retirant le **x** (exécution) au groupe (**g**), j'empêche désormais les autres chercheurs de lister le contenu ou d'entrer dans ce dossier spécifique.
2. **La validation :** Un dernier `ls -la` m'a permis de confirmer que la structure était enfin propre. Comme on le voit sur la capture, seul l'utilisateur principal conserve ses droits complets sur ce dossier sensible.

Résumé

Ce travail sur le `projet_répertoire` m'a permis de remettre les compteurs à zéro concernant la sécurité des accès. Concrètement :

- **Audit rapide** : J'ai utilisé `ls -la` pour ne rien rater, même les fichiers cachés comme `.project_x.txt`.
- **Nettoyage ciblé** : J'ai corrigé les permissions trop larges sur les fichiers de recherche pour qu'ils ne soient plus modifiables par n'importe qui.
- **Principe de sécurité** : Chaque étape a été guidée par le principe du moindre privilège : on donne l'accès nécessaire pour travailler, mais on verrouille tout le reste.

Au final, le répertoire est maintenant propre et sécurisé, ce qui permet à l'équipe de recherche de bosser sereinement sans risquer une fuite ou une modification accidentelle de leurs données.