320 Droits et utilisateurs

INF3173
Principes des systèmes d'exploitation

Jean Privat

Université du Québec à Montréal

Hiver 2021

Droits et utilisateurs

Utilisateurs (et groupes)

- `uid` : numéro d'utilisateur
- `gid` : numéro de groupe d'utilisateurs

$\rightarrow$ Pour le système vous n'êtes que des numéros

- uid $==$ 0 : super-utilisateur (`root`)

Noms des utilisateurs et groupes (Unix)

Le noyau gère pas les noms des utilisateurs et des groupes

- Fichiers `/etc/passwd` et `/etc/group`
- Fonctions `getpwuid(3)` et `getgrgid(3)`

Utilisateurs et processus

Paires d'identités

- Un utilisateur et un groupe d'utilisateurs = une paire
- Deux paires d'identités (réel et effectif) par processus
- pthreads partagent, fork hérite, exec préserve*
- `setuid(2)`, `setgid(2)`, `seteuid(2)`, `setegid(2)`

Sous Linux

- 4 paires distinctes (réel, effectif, sauvé, fichier†)
- Et des groupes supplémentaires
- `setresuid(2)`, `setfsuid(2)`, `setgroups(2)`, `credentials(7)`

*Sauf si setuid et/ou setgid, on y reviendra...

†Plus vraiment utilisé

Propriétaires des fichiers

Traditionnel Unix

- Chaque fichier du système possède
  - un numéro d'utilisateur propriétaire
  - un numéro de groupe propriétaire
- `chown(1)`, `chgrp(1)`, `chown(2)`
- Lors de la création d'un fichier :
  propriétaires = utilisateurs et groupes effectifs[‡]

Question

Pourquoi ça peut être un problème de stocker seulement les numéros d'utilisateur et groupes dans le système de fichiers ?

[‡] Sauf si setgid dans le répertoire, on y reviendra...

Rappel : droits traditionnels Unix

3 catégories d'accès (ugo)

- u (*user*/utilisateur) l'utilisateur propriétaire
- g (*group*/groupe) le groupe propriétaire
- o (*other*/autre) les autres

3 permissions par catégorie (rwx)

- `r` (*read*) : lire le contenu
- `w` (*write*) : modifier le contenu
- `x` (*execute*) : exécuter (si fichier) ou traverser (si répertoire)
- `chmod(1)`, `chmod(2)`

Questions

- Quels sont les droits nécessaires pour `stat(2)`? `chmod(2)`? suivre un lien symbolique? supprimer un fichier?
- Quand sont vérifiés les droits?

setuid (et setgid)

Bits supplémentaires du mode du fichier

- setuid: 4000, `u+s`
- setgid: 2000, `g+s`

Pour les fichiers exécutables

- Lors du `execve(2)`, l'utilisateur (et/ou groupe) effectif est changé pour celui du fichier
- RTFM pour les détails

→ **Question** Comment (et par qui) est contrôlée cette augmentation de privilèges ?

setgid pour les répertoires

- Sous Linux : un nouveau fichier héritera du groupe du répertoire (au lieu d'être le groupe effectif du processus)

Plus de droits sur les fichiers

Masque utilisateur

- Un par processus (threads partagent, fork hérite, exec préserve)
- Modifiée par `umask(2)`
- L'umask est retiré des droits des fichiers créés (`creat(2)`, etc.)

```
int creat(const char *pathname, mode_t mode);
```

```
droits_du_fichier = mode & ~umask
```

- Ne s'appliquent pas à `chmod(2)`, ni s'il y a des ACL par défaut

Encore plus

- ACL (*access control lists*): `acl(5)` → contrôle fin des droits
- MAC (*mandatory access control*)
  Exemples: `selinux(8)` et `apparmor(7)`