Version : **2020.01**

Dernière mise-à-jour : 2020/01/30 03:28

SO203 - Gestion des Droits

Dans sa conception de base, Solaris utilise une approche sécurité de type **DAC**. Cette approche est maintenue dans la mise en place et l'utilisation des **ACL** :

Type de Sécurité	Nom	Description
DAC	*Discretional Access Control*	L'accès aux objets est en fonction de l'identité (utilisateur,groupe). Un utilisateur peut rendre accessible aux autres ses propres objets.

Préparation

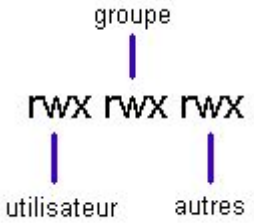
Dans le répertoire **/tmp**, créez un fichier tux.jpg grâce à la commande **touch**:

```
# cd /tmp;touch tux.jpg
```

> **Important** - Notez que le fichier créé est un fichier **texte**. En effet, Solaris ne tient pas compte de l'extension **.jpg**

Les Droits Unix Simples

Les autorisations ou droits d’accès en Solaris sont communiqués comme suit :



;#; ;#;

ou r = lecture, w = écriture et x = exécutable

Dans chaque inode est stocké le numéro de l'utilisateur à qui appartient le fichier concerné ainsi que le numéro du groupe. Quand le fichier est ouvert le système compare le numéro de l'utilisateur (UID) avec le numéro de l'utilisateur stocké dans l'inode (Utilisateur de Référence). Si ces deux numéros sont identiques, l'utilisateur obtient les droits du propriétaire du fichier. Si les numéros diffèrent, le système vérifie si l'utilisateur est dans le groupe référencé dans l'inode. Si oui, l'utilisateur aura les droits spécifiés pour le groupe. Si aucune condition n'est remplie, l'utilisateur se voit attribué les droits des «autres».

Les droits pour les répertoires sont légèrement différents :

r	Les éléments du répertoire sont accessible en lecture (lister)
w	Les éléments du répertoire sont modifiables (création et suppression).
x	Le nom du répertoire peut apparaître dans un chemin d'accès.

La Modification des Droits

Mode Symbolique

Afin de modifier les droits d'accès aux fichiers, on utilise la commande chmod dont le syntaxe est le suivant :

chmod [-R] catégorie opérateur permissions nom_du_fichier

ou

chmod [-R] ugoa +-= rwxXst nom_du_fichier

où

u	user
g	group
o	other
a	all
+	autorise un accès
-	interdit un accès
=	autorise exclusivement l'accès indiqué
r	read
w	write
x	execute
X	exécution si la cible est un répertoire ou si c'est un fichier est déjà exécutable pour une des *catégories* (ugo)
s	SUID/SGID bit
t	sticky bit

par exemple la commande suivante donnera aux autres l'accès en écriture sur le fichier tux.jpg :

```
# chmod o+w tux.jpg
# ls -l | grep tux.jpg
-rw-r--rw-   1 root     root           0 Jan 14 13:09 tux.jpg
```

Tandis que la commande suivante ôtera les droit d'accès en écriture pour l'utilisateur et le groupe :

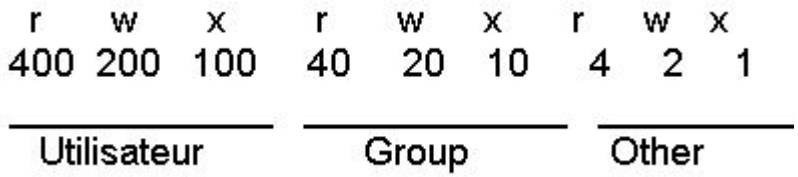
```
# chmod ug-w tux.jpg
# ls -l | grep tux
-r--r--rw-   1 root     root           0 Jan 14 13:09 tux.jpg
```

Important - Seul le propriétaire du fichier ou root peuvent modifier les permissions. Le droit de supprimer un fichier dépend des droits sur le répertoire dans lequel le fichier est

stocké et non des droits du fichier lui-même.

Mode Octal

La commande chmod peut également être utilisée avec une représentation octale (base de 8). Les valeurs octales des droits d'accès sont :



;#; ;#;

Important - Ainsi les droits rwx rwx rwx correspondent à un chiffre de 777.

La commande chmod prend donc la forme suivante:

```
chmod [ -R ] mode_octal nom_fichier
```

La commande suivante correspond donc à l'attribution des droits : rw- r– r–

```
# chmod 644 tux.jpg
# ls -l | grep tux
-rw-r--r--   1 root     root           0 Jan 14 13:09 tux.jpg
```

Les droits d'accès par défaut lors de la création d'un élément sont :

Répertoires	rwx rwx rwx	777

Fichier normal	rw– rw– rw-	666

La Commande umask

L’utilisateur peut changer ses critères lors de la création d’éléments en utilisant la commande umask.

Par exemple dans le cas où l’utilisateur souhaite que les fichiers créés dans le futur comportent des droits d’écriture et de lecture pour l’utilisateur mais uniquement des droits de lecture pour le groupe et pour les autres, il utiliserait la commande :

```
# umask 022 [Entrée]
```

avant de créer son fichier.

La commande umask sert à enlever des droits des droits maximaux :

Masque maximum lors de la création d’un fichier	rw- rw- rw-	666
Droits à retirer	— -w- -w-	022
Résultat	rw- r– r–	644

Dans l'exemple qui suit, on utilise la commande touch pour créer un fichier vide ayant les nouveaux droits par défaut :

```
# umask 044
# touch tux1.jpg
# ls -l | grep tux1
-rw--w--w-   1 root     root           0 Jan 14 13:15 tux1.jpg
```

Modifiez la valeur d'umask à 022 :

```
# umask 022
```

Modifier le propriétaire ou le groupe

La Commande chown

Dans le cas du fichier tux.jpg appartenant à root, celui-ci peut changer le propriétaire de root à l'utilisateur **nobody** avec la commande suivante :

```
# chown nobody tux.jpg
# ls -l | grep tux
-rw-r--r--   1 nobody   root          0 Jan 14 13:09 tux.jpg
-rw-r--r--   1 root     root          0 Jan 14 13:15 tux1.jpg
```

La Commande chgrp

Le même cas de figure s'applique au groupe :

```
# chgrp nobody tux.jpg
# ls -l | grep tux
-rw-r--r--   1 nobody   nobody        0 Jan 14 13:09 tux.jpg
-rw-r--r--   1 root     root          0 Jan 14 13:15 tux1.jpg
```

Ces deux commandes peuvent être combinées en une seule :

```
# chown root:root tux.jpg
# ls -l | grep tux
-rw-r--r--   1 root     root          0 Jan 14 13:09 tux.jpg
-rw-r--r--   1 root     root          0 Jan 14 13:15 tux1.jpg
```

Important - Le changement de propriétaire d'un fichier se fait uniquement par l'administrateur système **root**.

Les Droits Unix Etendus

SUID/SGID bit

Dans la sortie de la commande **ls -l /etc/passwd /usr/bin/passwd** ci-dessous, vous noterez que le fichier **/etc/passwd** possède les permissions **rw- r– r–** et qu'il appartient à **root**. Autrement dit **seul** root peut écrire dans ce fichier. Or, quand un utilisateur normal change son mot de passe, il écrit dans ce fichier. Ceci semble donc être une contradiction.

```
# ls -l /etc/passwd /usr/bin/passwd
-rw-r--r--   1 root     sys          802 Jan 14 12:55 /etc/passwd
-r-sr-sr-x   1 root     sys        26764 Jun 13  2012 /usr/bin/passwd
```

Pour remédier à cette apparente contradiction, Solaris dispose de deux droits d’accès étendus :

- Set UserID bit (SUID bit)
- Set GroupID bit (SGID bit)

Quand le SUID bit est placé sur un programme, l’utilisateur qui lance ce programme se voit affecté le numéro d’utilisateur du propriétaire de ce programme et ce pour la durée de son exécution.

Dans le cas donc du changement de mot de passe, chaque utilisateur qui lance le programme /usr/bin/passwd se trouve temporairement avec le numéro d’utilisateur du propriétaire du programme passwd, c’est à dire root. De cette façon, l’utilisateur peut intervenir sur le fichier /etc/passwd. Ce droit est indiqué par la lettre s à la place de la lettre x.

La même fonction existe pour le groupe à l’aide du SGID bit.

Pour assigner les droits, vous utiliserez la commande chmod :

- chmod u+s nom_du_fichier
- chmod g+s nom_du_fichier

En base huit les valeurs sont les suivants :

- SUID = 4000
- SGID = 2000

Inheritance Flag

Le SGID bit peut également être affecté à un répertoire. De cette façon, les fichiers et répertoires créés à l'intérieur auront comme groupe le groupe du répertoire parent. Ce droit s'appelle donc l'**Inheritance Flag** ou le **Drapeau d'Héritage**.

Par exemple :

```
# mkdir inherit
# chmod g+s inherit
# chgrp nogroup inherit
# touch inherit/test.txt
# mkdir inherit/testrep
# ls -l inherit
total 8
-rw-r--r--   1 root     nogroup        0 Jan 16 11:55 test.txt
drwxr-sr-x   2 root     nogroup      117 Jan 16 11:55 testrep
```

Sticky bit

Il existe un dernier cas qui s'appelle le **sticky bit**. Le sticky bit est utilisé pour des répertoires ou tout le monde a tous les droits. Dans ce cas, tout le monde peut supprimer des fichiers dans le répertoire. En ajoutant le sticky bit, uniquement le propriétaire du fichier peut le supprimer :

```
# chmod o+t /répertoire
# chmod 1777 /répertoire
```

Par exemple :

```
# mkdir /tmp/repertoire_public; cd /tmp; chmod 1777 repertoire_public
# ls -l | grep repertoire_public
```

```
drwxrwxrwt   2 root     root         117 Jan 14 13:25 repertoire_public
```

Les Droits Unix Avancés

Les ACLs

Au delà des droits étendus d'Unix, Solaris utilise un système d'ACL pour permettre une meilleure gestion des droits sur des fichiers.

Pour connaître les ACL positionnés sur un fichier, il convient d'utiliser la commande **getfacl** :

```
# cd /; mkdir rep; touch rep/fichier; cd rep; getfacl fichier

# file: fichier
# owner: root
# group: root
user::rw-
group::r--              #effective:r--
mask:r--
other:r--
```

Pour positionner des ACL sur un fichier, il convient d'utiliser la commande **setfacl** :

```
# setfacl -m user:nobody:rw- fichier
```

Important - L'option **-m** ajoute un ACL aux ACL existants.

Utilisez la commande **getfacl** pour visualiser le résultat :

```
# setfacl -m user:nobody:rw- fichier
```

```
# getfacl fichier

# file: fichier
# owner: root
# group: root
user::rw-
user:nobody:rw-         #effective:r--
group::r--              #effective:r--
mask:r--
other:r--
```

> **Important** - Bien que l'acl soit positionné, l'utilisateur nobody a un droit effectif de **r–**. Ceci est du au fait que le **mask** est **r–**. Le mask indique les permissions maximales qui peuvent être acccordées à un utilisateur ou un groupe tiers.

Il convient donc de changer la valeur du mask afin que nobody puisse récupérer le droit d'écriture :

```
# setfacl -m mask:rw- fichier
# getfacl fichier

# file: fichier
# owner: root
# group: root
user::rw-
user:nobody:rw-         #effective:rw-
group::r--              #effective:r--
mask:rw-
other:r--
```

Regardez maintenant l'effet des ACL sur un répertoire :

```
# mkdir rep1
# setfacl -s u::rwx,g::r-x,o:r-x,m:r-x,d:u::rw-,d:g::rw-,d:o:---,d:m:rw- rep1
# getfacl rep1

# file: rep1
# owner: root
# group: root
user::rwx
group::r-x              #effective:r-x
mask:r-x
other:r-x
default:user::rw-
default:group::rw-
default:mask:rw-
default:other:---
```

Important - Notez l'utilisation de la lettre **d** pour indiquer un droit par défault. L'option **-s** efface les anciens ACL et les remplace avec les nouveaux.

Créez maintenant un fichier appelé **fichier1** dans **/rep/rep1** et consulter les ACLs :

```
# touch /rep/rep1/fichier1
# getfacl rep1/fichier1

# file: rep1/fichier1
# owner: root
# group: root
user::rw-
group::rw-              #effective:rw-
mask:rw-
other:---
```

Noter que le fichier créé possède les droits par défaut positionnés sur le répertoire rep1.
