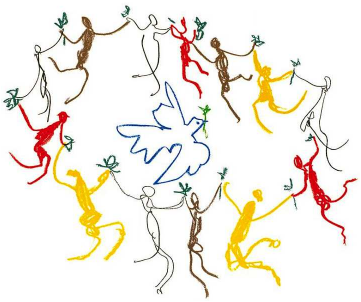
Sujet de TP - Organisation du partage

Remarques préliminaires

Les remarques constructives relatives à ce support sont à transmettre à `clot@univ-lyon1.fr`.

Unix est un système multi-tâche et multi-utilisateur. Traduction :
— multi-tâche : plusieurs programmes peuvent être exécutés en parallèle ;
— multi-utilisateur : plusieurs utilisateurs peuvent utiliser le système en parallèle.



Ces capacités engendrent la problématique du partage des ressources entre les différents programmes : tous les programmes sont, a priori [1], en concurrence pour l'utilisation des ressources, qu'ils soient associés au même login ou pas. La solution proposée par un système de type Unix repose sur le principe que tout est fichier, que chaque fichier est associé à un propriétaire et un groupe et que des droits sont définis pour le propriétaire, le groupe et les autres. Des extensions à ce modèle existent (e.g. les ACLs) comme cela est notamment le cas sur `minisfa`.

Dans une première partie, les commandes permettant de caractériser, en termes de permissions, les utilisateurs du système sont présentées. Dans la suite, nous introduisons les commandes permettant d'interagir avec les permissions relatives aux fichiers.

Connaître les droits associés à un login : W* R U ?

Dans un terminal de commande, il est important de toujours pouvoir répondre à trois questions :
— Où êtes-vous connecté ?
— Où êtes-vous positionné dans l'arborescence ?
— Quel login et quels groupes utilisez-vous ?

Sur un système donné, les réponses aux questions précédentes conditionnent les accès aux différentes ressources par le mécanisme de gestion des permissions ! Elles permettent de mieux cerner l'utilisation qu'il est possible de faire des ressources d'un système.

Ci-dessous, nous considérons d'abord la première question, puis la seconde.

- Ouvrez une première connexion dans laquelle vous resterez logué et une seconde connexion dans une autre console vers `minisfa`. Ceci vous permettra de tester les droits associés à votre login universitaire et ceux associés votre login personnel sur `minisfa`.
- Vous pouvez à tout moment connaître le login en cours d'usage dans une console à l'aide de la commande `whoami`. Testez la commande dans les deux consoles.
- La commande `who` permet d'obtenir diverses informations relatives aux utilisateurs logués sur le système. Après avoir déterminé le nom de votre console, utilisez `who` pour savoir si votre console accepte les messages. Il faut pour cela positionner une option que vous identifierez.
- En accord avec un de vos voisins, envoyez un message vers la console sur laquelle il est logué. Cet échange ne peut se faire qu'au sein d'un même système. Par conséquent, réalisez cet échange sur `minisfa` avec une commande que vous identifierez dans le manuel...

1. C'est le cas par défaut, mais il possible d'imaginer des programmes qui collaborent pour l'utilisation de ces ressources...

- Il est possible de modifier la gestion des messages par la console. Utilisez la commande `mesg` pour ne plus accepter de message dans votre console et testez ce nouvel état de la console.
- En quoi les informations données par `who` et `w` diffèrent ?
- A l'aide de la commande `id` dans chacun des environnements, relevez les groupes auxquels vos logins sont associés.
- Faites une recherche dans les pages du manuel sur le mot clé `group` et filtrez les lignes commençant par ce mot.
- Sur `minisfa`, comparez les sorties des commandes `id` et `groups`. Filtrez la sortie de la commande `id` afin d'obtenir la liste des groupes auxquels votre login est associé. Modifiez ce filtre afin chaque groupe apparaisse seul sur une ligne et que les lignes soient triées selon l'identifiant numérique du groupe.
- Modifiez la commande précédente afin de construire une chaîne construite comme suit :
  `-group GIDdeVotrePremierGroupe -o -group GIDSecond...-o -group GIDDernierGroupe`
  où, bien entendu, les identifiants de groupe correspondent aux groupes auxquels votre login est associé !
- A l'aide de commande `newgrp`, changez de groupe. Choisissez un des groupes additionnels auxquels votre login est associé. Contrôlez le changement à l'aide de la commande...Quelle commande ? Quittez ce groupe.
- Utilisez la commande `su` pour «devenir» `invite`. Vous demanderez en séance le mot de passe associé à ce login. Quels sont les groupes auxquels ce login est associé ? Donnez une commande par laquelle à partir de votre login vous auriez pu obtenir cette information.
- Selon les systèmes, l'exécution de `su` peut être limitée aux membres du groupe `wheel`. Vous essayerez également la commande `su` à partir de vos sessions locales (pas sur `minisfa`). En accord avec votre voisin, faites `su SonLogin` pour lequel il saisira le mot de passe. Le but est simplement de tester si la commande `su` est utilisable sur les machines de l'université avec vos permissions. En cas de succès, déconnectez-vous aussitôt.

Et maintenant, votre positionnement !

- Dans l'une des console, considérez le prompt. Vous donne-t-il une information sur votre position sur l'aborescence du système de fichier ?
- A l'aide de la commande `cd`, positionnez-vous dans le répertoire `/usr/share/`.
- Que réalise la commande `cd -` (oui oui, cédémoins) ?
- Affichez le contenu des variables `PWD` et `OLDPWD`. A quoi correspondent leurs valeurs ?
- Que permet de faire la commande `pwd` ?
- Que réalise la commande `cd ../../..` ?
- Que réalise la commande `cd ././.` ?
- Que réalise la commande `cd /usr/share/../share` ?
- Que réalise la commande `cd /..` ?
- Trouvez l'aide de la commande `cd`. Qu'est-ce que `cd` en réalité ?

Définir les permissions sur des fichiers

Domptage de `ls`

- Quel est le comportement de la commande `ls` utilisée sans argument ?
- A l'aide de `ls`, observez les permissions associées aux répertoires `~invite`, `~VotreLogin` et `~root`. Consultez l'aide associée à la commande `ls` afin d'obtenir cette information pour les répertoires mentionnés et seulement eux !
- Quelles informations relatives au répertoire `~invite` peut obtenir un usager quelconque du système ?
- Quelles informations relatives à votre répertoire personnel peut obtenir un usager quelconque du système ?
- Quelles informations relatives aux fichiers de votre répertoire personnel peut obtenir un usager quelconque du système ?
- Tout comme dans votre répertoire, le fichier `.bash_history` existe dans `~invite`. Quelles sont les permissions associées à ce fichier ? Pouvez-vous accéder au contenu de ce fichier ? Pourquoi ? Et le fichier `.bashrc` ? Pourquoi ?
- Le répertoire `~root` contient lui aussi un `.bash_history`. Quel espoir d'obtenir des informations liées à ce fichier un utilisateur peut-il avoir ?

Définir les permissions d'accès

- Quelles sont les permissions définies dans votre espace personnel ? Quels sont les répertoires lisibles par les autres membres du groupe auquel votre répertoire est associé (probablement `etudiant` si vous utilisez votre login d'étudiant de Lyon1) ? Vous pourrez filtrer les lignes produites par `ls` afin de ne retenir que celles associées aux répertoires.
- Creez un fichier (cmd `touch`) nommé `FichierVotreNomPrenom` et observez les permissions qui ont été positionnées par défaut. Affichez le masque de création en cours (en notation octale et symbolique).
- Calculez la somme des octets du masque et des octets des permissions (ce qui revient à faire un OU logique bit à bit).
- Modifiez les permissions de sorte à être le seul à pouvoir lire et écrire ce fichier (cmd `chmod`). Ajoutez une ligne à ce fichier que l'on suppose nommé `FichierVotreNomPrenom` par la commande :
`echo "une ligne" >> FichierVotreNomPrenom`
et vérifiez que le contenu du fichier a bien été modifié.
- Retirez-vous les droits de lecture et d'écriture sur ce fichier et assurez-vous de l'effectivité de ces nouveaux droits
- Rétablissez vos droits.
- Assurez-vous de savoir réaliser ces opérations en utilisant des notations octales ou symboliques.
- Utilisez la commande `mkdir` pour créer un répertoire nommé `RepDeTestVotreNomPrenom` et contrôlez ses permissions. Retirez les droits de lecture/écriture au groupe et aux autres.
- Déplacez le fichier créé précédemment dans ce répertoire (cmd `mv`) et retirez-vous les droits de lecture et d'écriture sur le répertoire.
- Testez à présent les différentes opérations possibles relatives à un répertoire : lecture, écriture et traversée.
- Rétablissez vos droits de lecture/écriture sur le répertoire et ajoutez des droits d'écriture, d'exécution et de lecture pour le groupe ou les autres, afin d'accorder à votre voisin ces permissions. Demandez à votre voisin de créer un fichier dans ce répertoire pour lequel il retirera toute forme de permission. Ayant droit d'écriture sur votre répertoire, vous êtes en droit de réaliser diverses opérations relatives à l'entrée associée à ce fichier. Tentez un renommage du fichier au sein du même répertoire, puis un déplacement vers votre «*home directory*» et enfin vers le répertoire `/tmp`. Comment justifiez-vous l'aboutissement et l'insuccès de chacune de ces opérations ? Pour

finir, supprimez ce répertoire.

- Créez un répertoire nommé `RepDeTestNomPrenom_2` pour lequel vous retirerez toutes les permissions du propriétaire, ajouterez toutes les permissions du groupe, changerez le groupe pour un groupe auquel vous appartenez ainsi que votre voisin (cmd `chgrp`).
- Tentez d'accéder à ce répertoire. Quelle raison explique que vous n'ayez pas accès à ce répertoire ?
- Tentez à présent de changer le propriétaire[2]. Il est probable que cette opération ne soit permisse qu'à `root` !
- Sollicitez votre voisin afin qu'il trouve un moyen d'acceder à ce répertoire.

Sticky bit

- Créez un nouveau répertoire nommé `RepDeTestStickyVotreNomPrenom` et positionnez le sticky bit, ainsi que les droits de lecture/écriture/exécution pour les autres. Vous vous assurerez de savoir réaliser cela à l'aide de la seule commande `mkdir`, ainsi qu'en couplant les commandes `mkdir` et `chmod`.
- Sollicitez à nouveau votre patient voisin afin qu'il crée un nouveau fichier auquel il retirera toutes les permissions.
- Tentez à présent de renommer son fichier, de le déplacer hors du répertoire, puis de l'effacer.
- Expliquez la possibilité de l'effacement observé (cf. manuel !).
- Sollicitez à nouveau votre patient voisin afin qu'il crée un nouveau fichier auquel il retirera toutes les permissions et qu'il sollicite un troisième acteur afin qu'il tente des opérations de renommage, déplacement et suppression du fichier créé.
- Supprimez votre répertoire en parametrant correctement la commande `rm`.

Usage de `umask`

- Quel est le masque de création pour vos fichiers, i.e. quelles sont les permissions implicites ?
- Quelles sont les permissions définies dans votre espace personnel ?
- Listez l'ensemble des répertoires en partant de votre `~` à l'aide de la commande `find`.
- Toujours à l'aide de cette commande, affichez les permissions associées à tous ces répertoires.
- Filtrez la sortie de la commande `ls -Rl ~` afin d'obtenir une sortie équivalente à la précédente.
- Utilisez la commande `wc` pour compter le nombre de répertoires présents dans l'arborescence débutant dans votre `~`.
- Comptez à présent les répertoires lisibles ou traversables par les autres.
- Avant de modifier les permissions relatives aux autres, vous allez rediriger la sortie de la commande `find` que vous avez élaboré afin de conserver l'information relative aux permissions sur l'ensemble de vos répertoires. Pour cela, ajoutez `> ~/ImageArboRep` à la suite de votre commande `find`...
- Contrôlez le fichier créé (i.e. son contenu !).
- Retirez à présent aux autres (mais pas au groupe) la possibilité de traverser et lire l'ensemble de vos répertoires. La commande `find` pourra vous aider dans cette tâche.

2. D'un système à l'autre, l'issue de cette commande peut varier. Tout dépend de la gestion choisie par le système. Notez que, en règle générale, les systèmes qui permettent l'opération de changement de propriétaire répositionnent à zero les bits SGID et SUID pour des raisons de sécurité triviales : sans ce mécanisme, un utilisateur peut programmer une suite d'operations dont le succès nécessite les permissions d'un autre utilisateur, permettre l'exécution à tous les utilisateurs, positionner le bit SUID et finalement changer le propriétaire par celui dont il cherche à acquerir les droits...Notez toutefois, comme indiqué en cours, l'exploitation du SUID et du SGIG n'est pas immédiate !


Après avoir réalisé la commande `ls -R ~ > ~/ImageArboFull`, retirez à présent aux autres (mais pas au groupe) la possibilité de traverser et lire l'ensemble de vos fichiers en utilisant la commande `chmod` de façon récursive.


Modifiez votre masque de création de fichier afin que les autres n'aient plus de permission par défaut. Après cette opération, logguez-vous dans une autre console et testez votre masque.

Retour sur la recherche d'information


Proposez un script filtrant la sortie de `find ~ -type d -printf '%m %g %h/%f\n'` afin de retenir les lignes relatives aux répertoires correspondant à une permission de traversée et de lecture par les membres du groupe. Seule l'information relative au nom du répertoire et au nom du groupe sera conservée.


Proposez un script filtrant la sortie standard de `find ~ -type d -printf '%m %g %h/%f\n' | sort -k 3` afin de retenir les lignes relatives aux répertoires correspondant à une permission de traversée et de lecture par les membres du groupe. Par ailleurs, parmi ces lignes, seules celles correspondant à un répertoire atteignable et lisible par le groupe seront conservées. Cette information est plus fine car même si un répertoire est lisible, si un répertoire parent n'est pas traversable, alors le premier ne peut être atteint !


Comment déterminer les fichiers de votre répertoire personnel qui sont exécutables par vous, dont le propriétaire est vous-même ou `root` et qui ont été modifiés il y a au plus trois jours ?


Proposez une commande qui recherche toutes les entrées à partir de votre home qui sont des fichiers exécutables (mais pas des liens), et qui produit une sortie formatée donnant pour chaque entrée le nom du fichier, sa date de dernière modification et le répertoire où il a été trouvé.