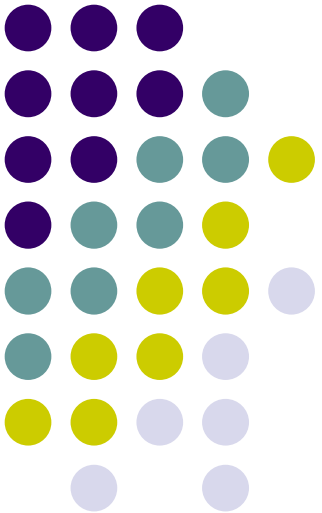
Droits et permissions sous Linux

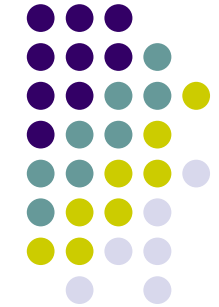
Philippe.Arnould@univ-pau.fr



Plan

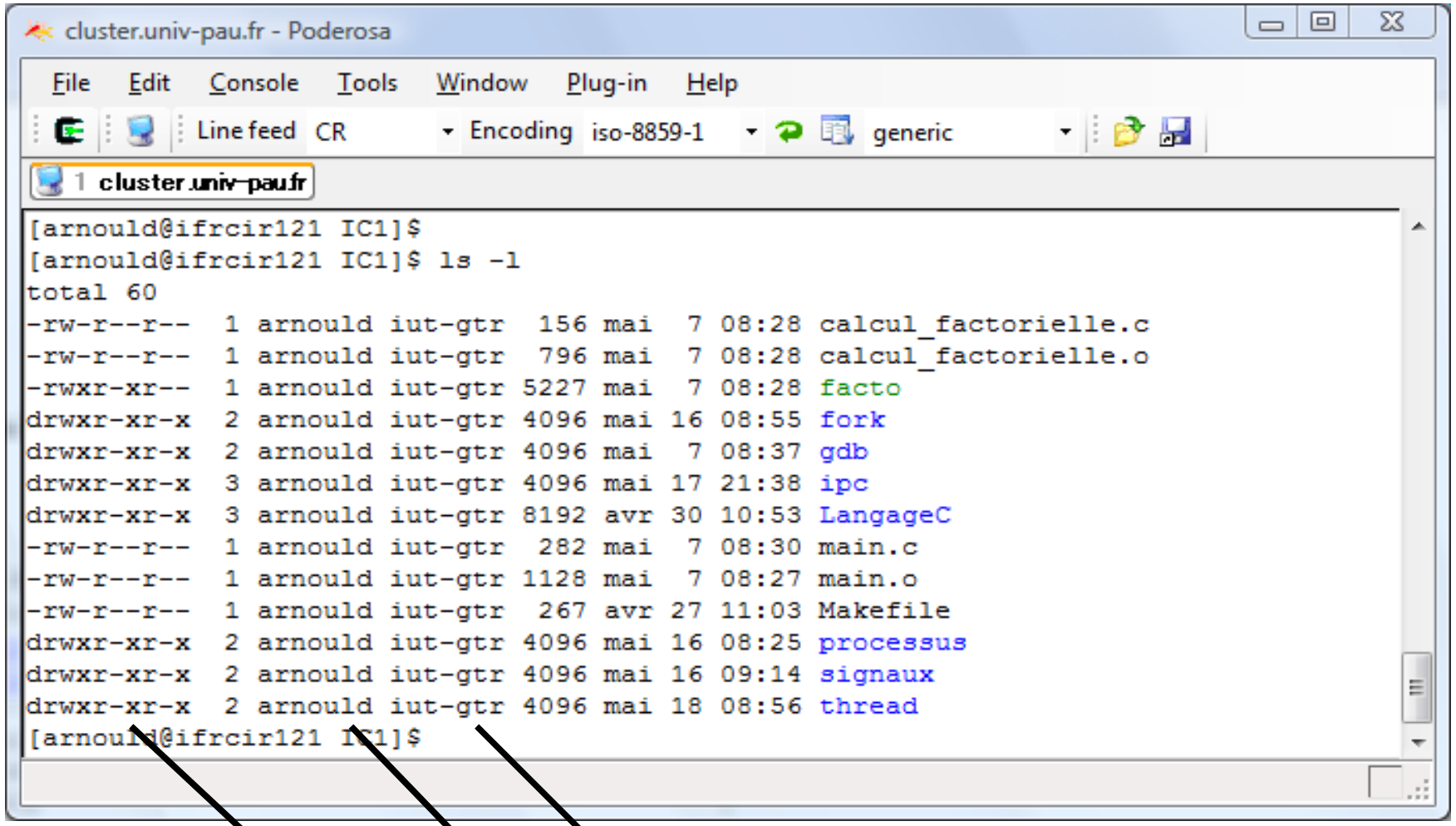
- Droit d’accès aux fichiers
- Changement de propriétés
  - Changement de propriétaire
  - Changement de groupe propriétaire
  - Changement de droit des fichiers
- Les droits spéciaux

Droit d’accès aux fichiers



- Linux est un sytème multi-utilisateurs. De ce fait, tout le monde ne peut pas tout faire, excepté l’administrateur (root), qui à le droit de lire et d’écrire sur tous les fichiers et tous les répertoires, ainsi que d’executer n’importe quelle tâche inhérente au système. Il faut donc établir des règles et a fortiori donner des privilèges à certains utilisateurs et en priver à d’autres.

Droit d'accès aux fichiers

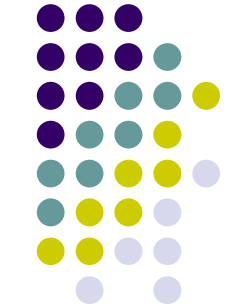


Groupe du propriétaire
Propriétaire du fichier
Droits du fichier

Droit d'accès aux fichiers

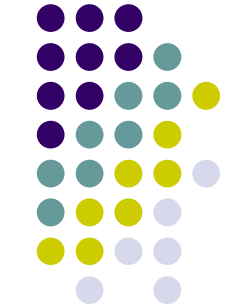
- Regardons la première colonne. Celle ci fournit des informations sur les droits :
  - le premier caractère de cette colonne est un d, un - ou un l :
    - d signifie qu'il s'agit la d'un répertoire (exemple : signaux est un répertoire),
    - - signifie qu'il s'agit d'un fichier(example : Makefile est un fichier)
    - l un lien
  - ensuite il y a trois groupements de trois caractères (rwx,r—,r-x, etc ...) soit neuf caractères au total : ce sont les **groupements de permission**
    - pour un fichier r signifie readable(en lecture), w writable(en écriture) et x executable
    - pour un répertoire r signifie readable(lire le contenu du répertoire avec ls ou dir), w writable (création, déplacement et suppression de fichiers) et x executable (accès aux fichiers commande cd)

Droit d'accès aux fichiers



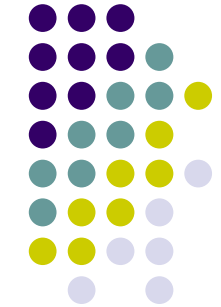
- Par exemple, r-x signifie que le fichier est en lecture et executable mais pas en écriture. Un fichier possèdant l'attribut rw- est en lecture et en écriture mais pas executable, etc... Le premier groupement correspond au droit du propriétaire (user), le second au droit du groupe propriétaire (group) le troisième aux autres(others). Le fichier facto est
  - rwx lecture + écriture + executable pour le propriétaire(user)
  - r-x lecture + executable pour le groupe(group)
  - r-x lecture + executable pour les autres(others).
- Le système de fichiers est agencée de telle manière à ce qu'un fichier ou répertoire appartienne à un utilisateur (propriétaire) et à un groupe. Tout fichier possède trois groupements :
  - u user, c'est à dire le propriétaire
  - g group le groupe
  - o others et les autres.

Changement de propriétaire



- La commande chown (change file owner) permet de changer de propriétaire. Seul le propriétaire actuel du fichier ou du répertoire peut lancer cette commande. À noter évidemment que le super-utilisateur root possède tous les droits sur tous les fichiers ! On l'utilise comme suit pour un fichier:
  - chown nouvel_utilisateur nom_de_fichier
- Pour changer le propriétaire d'un répertoire et de ses sous-répertoires, on utilise l'option -R :
  - chown -R nouvel_utilisateur nom_de_repertoire

Changement de groupe propriétaire



- La commande chgrp permet de changer de groupe propriétaire. Ce changement de groupe peut être opéré par le super utilisateur root ou par le propriétaire lui même si et seulement si ce dernier est membre du groupe.
- chgrp fonctionne comme suit pour les fichiers
  - chgrp nouvel_utilisateur nom_de_fichier
- et comme cela pour les répertoires
  - chgrp -R nouvel_utilisateur nom_de_repertoire
- On peut évidemment vouloir combiner les deux actions précédentes c'est à dire modifier le propriétaire et le groupe propriétaire de manière simultanée. On utilise la commande chown, pour un fichier cela donne
  - chown nouvel_utilisateur.nouveau_groupe nom_de_fichier
- pour un répertoire
  - chown -R nouvel_utilisateur.nouveau_groupe nom_de_fichier

Changement de droit des fichiers

- Le changement de droit des fichiers peut être opéré évidemment par le super-utilisateur root (il peut tout faire !!!) et par le propriétaire lui-même. On distingue trois type de modifications concernant les droits des fichiers :
  - + l'ajout de droits
  - - la suppression de droits
  - = la fixation de droits
- On distingue également trois types d'utilisateurs :
  - u user, c'est à dire le propriétaire
  - g group le groupe
  - o others et les autres
- et leur regroupement est défini par
  - a all, tout les utilisateurs.
- Les trois types de modifications +,-,= seront opérés sur les sur les groupes u,g,o,a en octroyant les droits r,w,x.

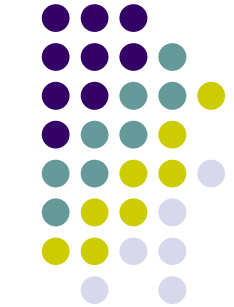
Changement de droit des fichiers

- Pour modifier les droits d'un fichier on utilise chmod, comme suit :
  - chmod utilisateur modification droit(s) nom_du_fichier
- ou si vous préférez
  - chmod [u g o a] [+ - =] [r w x] nom_du_fichier
- Pour les répertoires on procède comme précédemment en ajoutant l'option récursive -R :
  - chmod -R [u g o a] [+ - =] [r w x] nom_du_repertoire

Changement de droit des fichiers

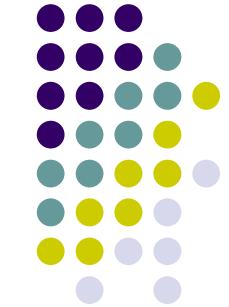
- On peut vouloir changer les droits de fichiers ou de répertoires non pas avec l'attribut [u g o a] [+ - =] [r w x] mais en fixant un attribut octal [0-7 0-7 0-7] .... Ca va être plus clair :
  - --- 0->000----->aucun droit
  - --x 1->001----->execution
  - -w- 2->010----->écriture
  - -wx 3->011----->écriture-execution
  - r-- 4->100----->lecture
  - r-x 5->101----->lecture-execution
  - rw- 6->110----->lecture-écriture
  - rwx 7->111----->lecture-écriture-execution
- On fonctionne en base octale (numérotation de 0 à 7), la correspondance en base binaire est assez évidente.
- `arnould@cluster:~/test $ chmod 770 fonction.php`
- `arnould@cluster:~/test $ ls -l`
- `total 0`
- `-rwxrwx---  1 arnould arnould 0 2005-12-16 18:08 fonction.php`

Les droits spéciaux



- SETUID
- Regardons les droits du fichier /usr/bin/passwd
- `arnould@cluster:~/test $ ls -l /usr/bin/passwd`
- `-rwsr-xr-x 1 root bin 25648 2005-10-11 18:14 /usr/bin/passwd`
- Il apparait un s. Cela signifie que le fichier est setuid. Etre setuid signifie que lorsque le programme est exécuté, il est avec le droit du propriétaire. Dans l'exemple précédent, lorsque le groupe bin excute la commande passwd, durant cette execution les membres de bin auront les droits de root.
- Quand un fichier exécutable est propriété de l'administrateur (root), et est rendu setuid, tout processus exécutant ce fichier peut effectuer ses tâches avec les permissions associées au root, ce qui peut constituer un risque de sécurité pour la machine, s'il existe une faille dans ce programme. En effet, un hacker ou un cracker pourrait utiliser cette faille pour s'arroger des droits d'administrateur et effectuer des opérations réservées, par exemple se créer un compte d'accès illimité en temps et en pouvoirs.
- Notez qu'il est parfaitement possible de rendre un programme setuid d'un compte non-root, ce qui diminue nettement les risques.
- Pour d'évidentes raisons de sécurité, un processus ne peut être tracé par un utilisateur non privilégié au travers de l'exécution d'un programme setuid. Toute tentative de traçage d'un tel processus se traduit généralement par la simple non-application du changement d'utilisateur que prévoyait le bit setuid.

Les droits spéciaux



- SETGID
- De manière identique, un exécutable est setgid, et s'exécuter avec les droits du groupe auquel il appartient.
- STICKY
- Un utilisateur ayant les droits d'écriture au sein d'un répertoire peut effacer n'importe quel fichier de ce répertoire. Cela peut s'avérer dangereux dans le cas où un répertoire est partagé par plusieurs personne. De ce fait, on introduit le sticky bit et dès lors l'utilisateur ne peut effacer que les fichier qui lui appartiennent. C'est le cas par exemple du répertoire /tmp.
- Quand on écrit les permissions en octal, setuid, setgid et sticky bit sont représentés par une nouvelle série de 3 bits, qui se place avant les 3 autres séries: setuid=4, setgid=2, sticky=1. Ainsi, sur ma machine, le serveur de mail /usr/sbin/sendmail a les droits rwsr-sr-x (rwxr-xr-x, setuid, setgid); en octal, ça donne 6775.