

Installation et configuration des systèmes et réseaux

Jean-Christophe ELINEAU



Directeur cluster open source
Nouvelle-Aquitaine : **Aquinetic**

Co-Président
PARIS OPEN SOURCE SUMMIT 2016

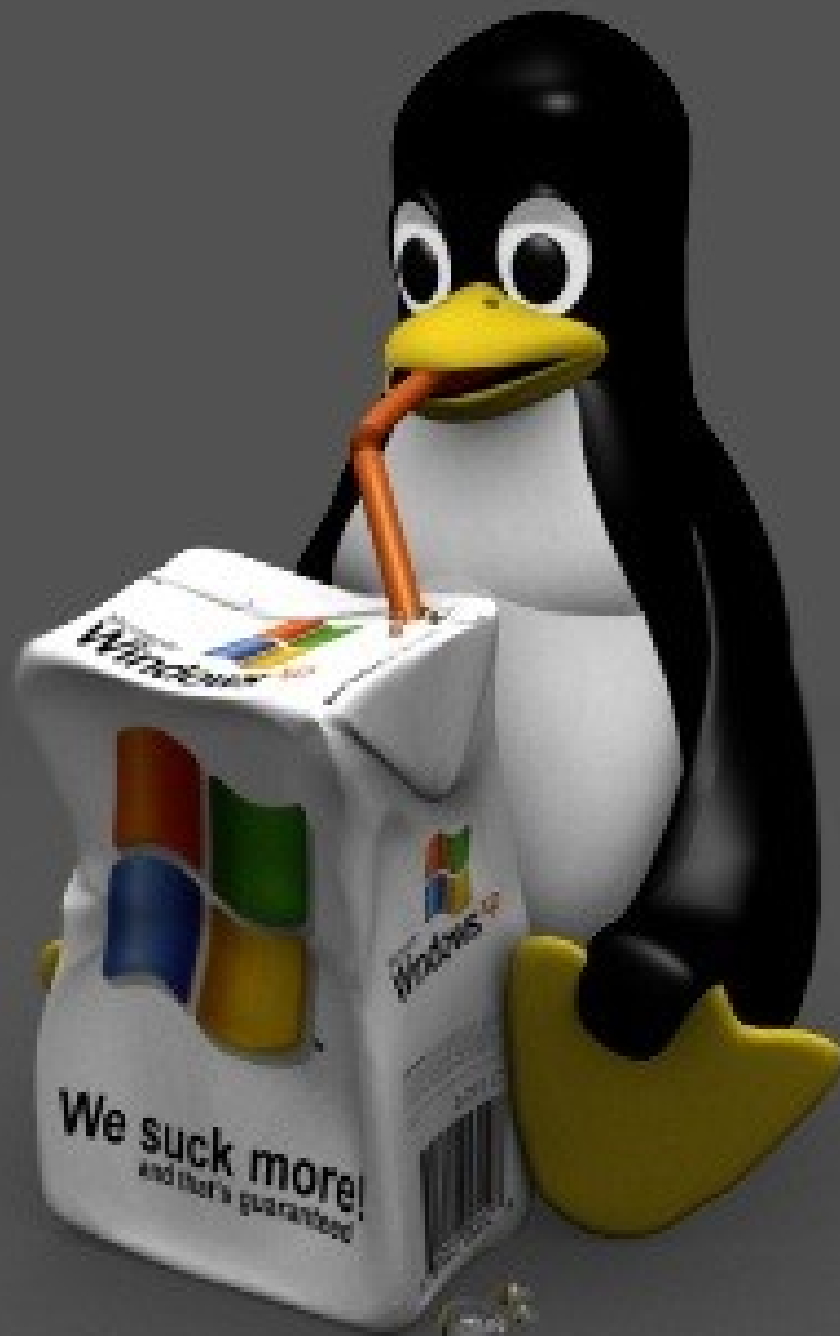
jc.elineau@pole-aquinetic.fr

Objectifs

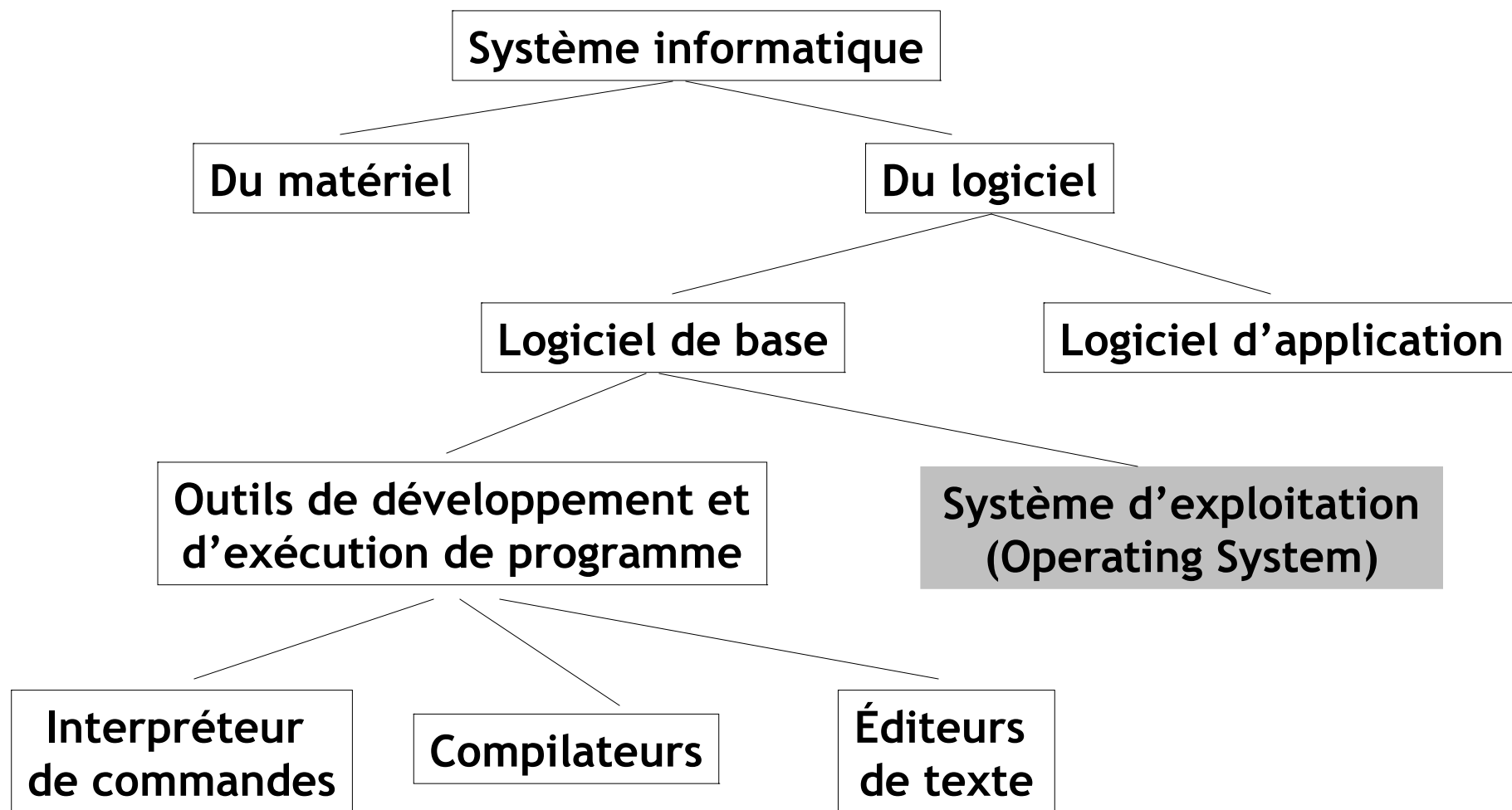
- Pouvoir installer et configurer des machines sous Linux et Windows.
- Pouvoir installer, configurer et maintenir un serveur sous linux: OS, DHCP, DNS, Postfix, NTP, LDAP, SAMBA
- Être capable de configurer un réseau local utilisant des postes sous Linux et Windows.
- Prendre en compte les problèmes d'authentification (chiffrement, PAM, LDAP, etc.)

1. Généralités
2. L'O.S. Linux : historique
3. Linux :
4. Les processus
5. Compression / décompression
6. Utilisation du réseau
7. Les éditeurs de texte

1. Généralités



Un système informatique ?



Outils de développement et d'exécution de programme

- **L'interpréteur de commandes (shell)** : permet d'accéder aux fonctions du système à l'aide d'un *langage de commande*
- **Les compilateurs** : sont chargés de traduire des programmes écrits dans des langages de haut niveau en une suite d'instructions en *langage machine*
- **Les éditeurs de textes** : permettent de saisir et modifier du texte (par exemple des programmes)

Un système d'exploitation ?

- Un **système d'exploitation** est une ensemble de procédures manuelles et automatiques qui permet à un groupe d'utilisateurs de partager efficacement un ordinateur. C'est aussi un ensemble de procédures cohérentes qui a pour but de gérer la pénurie de ressources
- Un **système d'exploitation** est un ensemble de programmes et de fonctions conçus pour faciliter et optimiser l'utilisation des unités physiques de l'ordinateur
- Le seul programme qui tourne constamment dans une machine
- Il existe plusieurs systèmes d'exploitation. Ils varient selon :
 - le type de matériel
 - la complexité des tâches à effectuer
 - les logiciels qu'ils doivent supporter

Le rôle d'un système d'exploitation

- Fournir à l'utilisateur l'équivalent d'une machine étendue ou virtuelle plus simple à programmer que la machine réelle
- Gérer de manière équitable et optimale l'allocation des processeurs, de la mémoire et des périphériques aux différents programmes qui les sollicitent
- Exemples de systèmes d'exploitation :
 - Mac-OS (le système Macintosh)
 - Windows (NT, 95, 98)
 - Unix, Linux
 - etc.

2. L'O.S. Linux : Historique

Deux incontournables



- ***Richard Stallman 1984***
- ***Linus Torvalds 1991***



Stallman



- La FSF (*Free Software Foundation*) a été fondée au début des années 80 par Richard M. Stallman, chercheur au laboratoire d'Intelligence Artificielle du MIT. Le but de cette fondation est de développer des logiciels *libres*.
- Le projet GNU est un projet de la FSF dont le but est de développer un système d'exploitation complet, distribué selon les conditions de la GPL.

Torvalds



- En 1991, Torvalds, ne voulant pas être limité par MS/Dos, a tout d'abord utilisé un clone d'Unix appelé Minix lui-même limité, il décida de réécrire le code.
- Torvalds utilisa ses propres travaux avec ceux de ses professeurs et les ajusta pour qu'ils fonctionnent avec les nombreux programmes déjà existants pour le projet GNU et presque par accident il accomplit l'exploit de faire un OS ayant la stabilité, la puissance et la flexibilité de Unix le tout sur un processeur 386 et 4 Mo Ram.

Torvalds

Deux facteurs complèteront le travail de Torvalds

Dans un premier temps

La sortie de Linux coïncide avec l'avènement populaire du siècle l'Internet.

Dans un deuxième temps

Cette période coïncide également avec une baisse fulgurante des **prix** d'ordinateur.

3. Linux :

C'est quoi Unix ?

- Unix est né au début des années 70 dans les laboratoires Bell
- Unix est un système :
 - **multi-utilisateurs** : plusieurs personnes peuvent partager les ressources de la même machine
 - **multi-tâches** : plusieurs programmes ou logiciels peuvent s'exécuter concurremment
- Il existe plusieurs versions commerciales :
 - AIX de IBM
 - Sun Solaris de SUN Microsystems
 - HP-UX de Hewlett Packard
 - etc.
- Plusieurs versions d'UNIX sont nées pour PC :
 - Linux
 - FreeBSD
 - OpenBSD
 - NetBSD ...

Organisation du système Unix

Le système Unix est organisé en couches :

- **Noyau** : la couche de plus haut niveau, elle assure la communication avec le matériel. Le noyau s'occupe de :
 - la gestion de la mémoire,
 - l'accès aux périphériques (disque dur, lecteur de CD-Rom, clavier, souris, ...),
 - la gestion du réseau, ...
- **Shell** : interprète les ordres de l'utilisateur et les fait exécuter par le noyau. Les ordres peuvent être passés soit directement au clavier, soit en utilisant des outils graphiques de plus haut niveau
- **Applications** : interagissent avec l'utilisateur ou avec d'autres applications et communiquent avec le shell ou avec le noyau

C'est quoi Linux ?

- Nous avons l'habitude de dire que Linux est un système d'exploitation
- Linux n'est en réalité que le noyau du système d'exploitation
GNU/Linux (GNU = GNU is Not Unix)
- GNU est un projet qui a apporté des tas d'utilitaires au noyau Linux :
 - compilateur gcc,
 - tar, man, bash ...

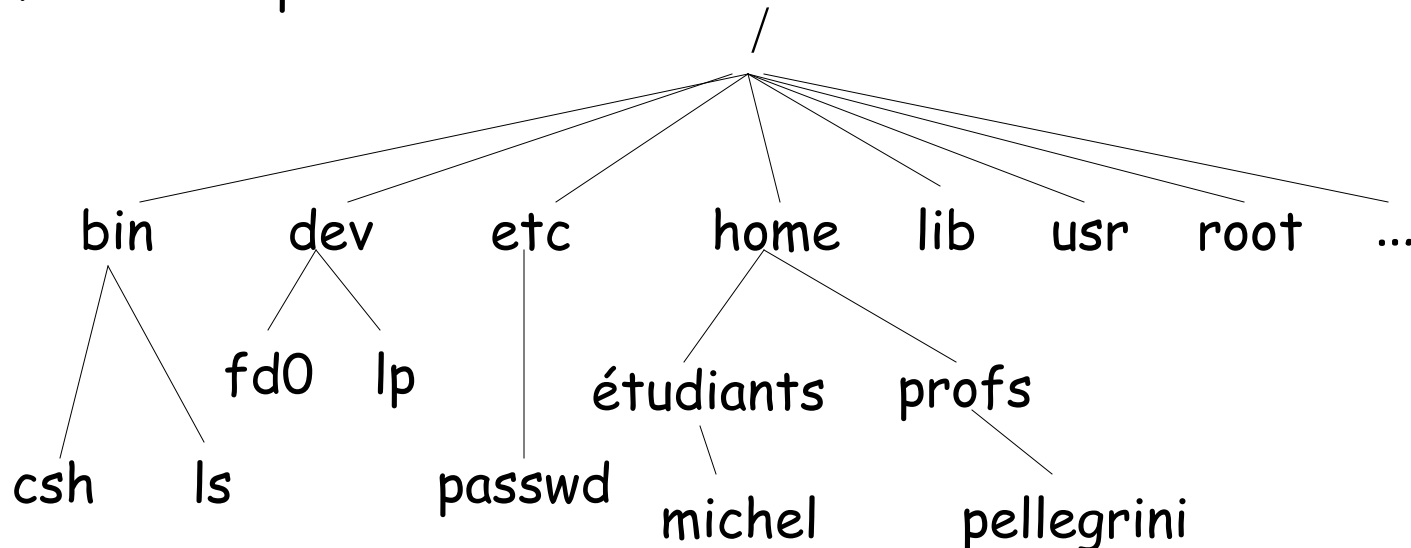
Linux est un noyau
GNU est un ensemble de programmes utilitaires
GNU/Linux est le système d'exploitation

Quelques liens :

- Histoire d'Unix : <http://www.bell-labs.com/history/unix/>
- Page officielle de Linux : <http://www.linux.org>
- Page officielle de GNU : <http://www.gnu.org>
- Histoire de Linux : <http://www.linux10.org/history/>

La hiérarchie des répertoires

- Linux définit un système de fichiers hiérarchique avec un certain nombre de répertoires standards



- `/root` est le répertoire d'accueil du super-utilisateur (administrateur)
- `/bin` contient généralement les programmes utiles au démarrage
- `/etc` contient les fichiers de configurations
- `/dev` contient les fichiers relatifs aux devices (périphériques)
- `/home` contient les répertoires des utilisateurs
- `/lib` contient les librairies du système
- `/usr` contient les programmes ajoutés au système

Se logger

- Linux possède un mécanisme d'identification connu sous le nom de **login**
- Pour utiliser un système Linux sur une machine, il faut avoir un **compte** sur cette machine
- Pour se connecter sur une machine il faut rentrer au clavier :
 - son nom d'utilisateur : **login**
 - son mot de passe : **password**
- Le système vérifie la correspondance entre le login et le mot de passe
 - si échec, il refuse l'accès
 - si correct, il lance la procédure de login (analyse différents fichiers de configuration et met en place l'environnement de l'utilisateur)
- L'utilisateur est alors placé dans son répertoire d'accueil : c-à-d
/home/pellegrini

Changer son mot de passe

- Si vous souhaitez changer votre mot de passe, la commande pour réaliser cette opération est : **passwd**

% passwd

Changing password for USER on MACHINE

Old password:

--entrez votre mot de passe courant

New password:

--entrez votre nouveau mot de passe

Retype new password:

--rentrez votre mot de passe

Entry has changed on filemon

Quel Shell ?

- Après le login, l'utilisateur accède à un interpréteur de commandes ou shell
- Le shell affiche un «prompt» et attend les commandes de l'utilisateur
- Il en existe plusieurs avec des fonctionnalités et des interfaces différentes les uns des autres
 - sh : Bourne Shell (shell standard)
 - ksh : Korn Shell
 - csh : C Shell
 - bash : GNU (Bourne Again Shell)
- Pour savoir quel shell est utilisé, tapez :

```
% echo $SHELL  
/bin/bash
```

- Le liste des shells autorisés : /etc/shells

Quelle est mon identité ?

- Pour Linux, l'identité d'un utilisateur est celle sous laquelle il se logge
- La commande **whoami** vous donne votre identité

```
% whoami  
pellegrini
```

- L'utilisateur appartient également à un ou plusieurs groupes
- La commande **id** vous donne votre identité et votre groupe

```
% id  
uid=5230(pellegrini) gid=64(profs) groups=64(profs)
```

n° de l'utilisateur

utilisateur

n° du groupe

groupe

Premières commandes : pwd et ls

- Une commande est un mot-clé avec éventuellement des options

```
% commande -options arguments
```

- La commande **pwd** (print working directory) indique le répertoire courant

```
% pwd  
/home/profs/pellegrini
```

- La commande **ls** permet d'afficher le contenu d'un répertoire

```
% ls  
Cours.tex   Examen_Linux.pdf  Recherche  Tps_Linux
```

Commande : ls avec options

- Avec l'option -l (pour version longue) plus d'informations sont affichées

```
% pwd
/home/pellegrini
% ls -l
-rw-r----- 1 pellegrini profs 362514 Sep 5 12:40 Cours.tex
-rwxrw-r-- 1 pellegrini profs 1024 Sep 1 2:10 Examen_Linux.pdf
drw-r--rw- 4 pellegrini profs 10 Jan 7 15:41 Recherche/
drwxrwxrwx 6 pellegrini profs 8425 Mar 2 11:38 Tps_Linux/
```

- ls sur un fichier affiche le nom de ce fichier si celui ci existe

```
% ls Cours.tex
Cours.tex
```

Commande : ls avec options

- Les principales options sont :
 - -l : format détaillé
 - -a : liste aussi les fichiers qui commencent par « . »
 - -d : si l'argument est un répertoire, la commande liste seulement son nom et pas les fichiers qu'il contient
 - -t : affiche en triant par date de dernière modification
 - -g : affiche les informations sur le groupe

Autorisation d'accès	propriétaire	Taille du fichier	Nom du fichier
drwxrwxrwx	6	8425	Tp_Linux/
	pellegrini	Mar 2 11:38	
	profs		
Type du fichier	Nb de liens	groupe	date de dernière modification

Type du fichier

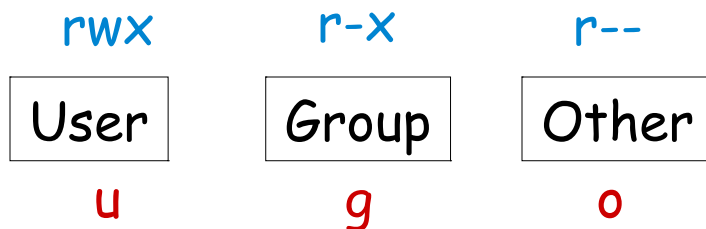
- L'indicateur du type de fichier peut prendre :
 - - : un fichier ordinaire
 - d : un répertoire
 - l : un lien symbolique
 - b : un fichier spécial de type bloc (périphériques ...)
 - c : un fichier spécial de type caractère (périphériques ...)
 - s : socket
 - ...

Droits d'accès aux fichiers

- Les fichiers possèdent un certain nombre d'attributs qui définissent les autorisations d'accès.

r autorisation à lire : read
w autorisation à écrire : write
x autorisation à l'exécution : execute

- Ces attributs sont groupés en 3 groupes de 3 attributs



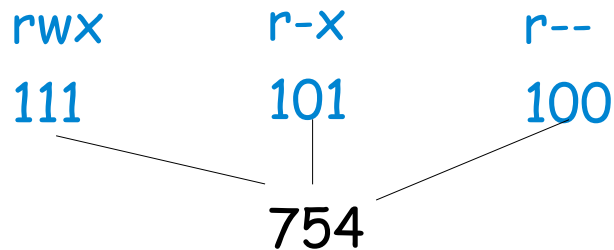
- rwxr-xr-- : fichier ordinaire : lecture, écriture et exécution permise pour le propriétaire, lecture et exécution pour le groupe et seulement lecture pour les autres. Il est donc impossible aux membres du groupe et aux autres utilisateurs d'écrire dans ce fichier

Modification des droits d'accès

- La protection d'un fichier ne peut être modifier que par le propriétaire
- La commande utilisée est : **chmod** (Change MODe)
- Le mode d'utilisation de cette commande est le suivant :

Par un nombre octal

% chmod [nombre octal] fichier



Représentation binaire

$$7 = 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0$$

$$5 = 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$$

Droits d'accès aux répertoires

- L'interprétation des droits est différente de celle des fichiers
- Les informations concernant un répertoire est données par la commande : **ls -dl répertoire**
- L'interprétation des protections est :
 - r : autorise la lecture du contenu du répertoire, permet de voir la liste des fichiers (et sous-répertoires) contenu dans le répertoire.
 - x : autorise l'accès au répertoire (à l'aide de la commande cd).
 - w : autorise la création, la suppression et le changement du nom d'un élément du répertoire. Cette permission est indépendante de l'accès aux fichiers du répertoire.

Se déplacer dans l'arborescence

- La commande permettant de se déplacer dans une arborescence est :
cd répertoire (change directory)

```
% pwd  
/home/profs/pellegrini  
% cd Enseignement  
% pwd  
/home/profs/pellegrini/Enseignement
```

- Chaque répertoire contient 2 entrées supplémentaires :
 - « . » : désigne le répertoire courant
 - « .. » : désigne le répertoire parent
- On peut se déplacer en utilisant un chemin : **cd chemin**
- Deux types de chemins : **absolu** ou **relatif**

Se déplacer dans l'arborescence

- Chemin absolu : chemin qui part directement du répertoire racine

```
% pwd  
/home/profs/pellegrini  
% cd /home/profs/pellegrini/Enseignement  
% pwd  
/home/profs/pellegrini/Enseignement
```

- Chemin relatif : chemin qui part du répertoire courant

```
% pwd  
/home/profs/pellegrini  
% cd Enseignement  
% pwd  
/home/profs/pellegrini/Enseignement
```

Commandes liées aux répertoires

- La commande servant à créer des répertoires est :
mkdir [options] répertoires... (make directory)
- Il suffit d'avoir le droit d'écrire (w) dans le répertoire père

- Pour créer une arborescence entière, on utilise l'option -p
Exemple : créer l'arborescence ~/TP_Linux/TP_Groupe1
mkdir -p TP_Linux/TP_Groupe1

- La commande servant à supprimer un répertoire est :
rmdir repertoire (remove directory)
- Le répertoire doit être vide

Commandes liées aux répertoires

- La commande servant à copier un fichier d'un répertoire vers un autre répertoire :

`cp fichier_source repertoire_destination` (copy)

- Pour copier des fichiers dans un répertoire :

`cp -i fichiers... repertoire_destination`

- La commande servant à copier tous les fichiers d'un répertoire :

`cp -r repertoire_source repertoire_destination`

- Toute l'arborescence du répertoire source est copiée dans le répertoire destination

- Les nouveaux fichiers se trouvent dans le répertoire :

`repertoire_destination/repertoire_source`

La recherche d'un fichier : find

- La commande **find** parcourt les répertoires et leurs sous-répertoires de manière récursive à la recherche de fichiers

- La syntaxe de cette commande est :

find répertoire(s) critère_de_sélection option(s)

- Un répertoire ne peut être parcouru que si l'utilisateur dispose des droits de lecture et d'exécution sur ce répertoire ou sous-répertoire

- Options de sélection des fichiers et répertoires :

- -print : affiche le chemin d'accès pour chaque fichier trouvé
- -name : recherche par nom de fichier
- -type : recherche par type de fichier
- -user : recherche par propriétaire
- -size : recherche par taille du fichier
- -atime, -amin : recherche par date de dernier accès (jour, minute)
- -mtime, -mmin : recherche par date de dernière modification (jour, minute)
- -ctime, -cmin : recherche par date de création (jour, minute)
- -perm : recherche par autorisation d'accès

La recherche d'un fichier : find

```
% find . -type d -print
```

Affiche tous les répertoires contenus dans le répertoire courant

```
% find . -type f -name '*s*' -print
```

Affiche tous les fichiers contenus dans le répertoire courant et dont le nom contient la lettre s

```
% find . -type f -size +200k -print
```

Affiche tous les fichiers de plus de 200 Ko

```
% find . -mtime -3 -print
```

Affiche tous les fichiers dont la date de la dernière modification remonte à moins de trois jours

```
% find /home/TP_Linux -type d -perm 755 -print
```

Affiche tous les sous-répertoires du répertoire /TP_Linux ayant comme autorisations d'accès rwxr-xr-x

La recherche d'un mot : grep

- La commande **grep** permet de rechercher, dans un ou plusieurs fichiers, toutes les lignes qui contiennent une chaîne de caractères donnée
- La syntaxe : **grep option(s) expression fichier(s)**
- Les options :
 - **-n** : fait précéder chaque ligne affichée par son numéro de ligne dans le fichier source
 - **-v** : affiche toutes les lignes sauf celles contenant **expression**
 - **-l** : n'affiche que les noms des fichiers dont au moins une ligne satisfait à la recherche
 - **-i** : ne fait aucune distinction entre les majuscules et les minuscules
 - **-c** : affiche le nombre de lignes qui contiennent l'expression

La recherche d'un mot : grep

% grep read programme.c

Affiche toutes les lignes du fichier **programme.c** contenant **read**

% grep -n read programme.c

Affiche avec la numérotation toutes les lignes du fichier **programme.c** contenant **read**

% grep -i 'else do' programme.c

Affiche toutes les lignes du fichier **programme.c** contenant la chaîne de caractères **'else do'** en majuscules ou minuscules

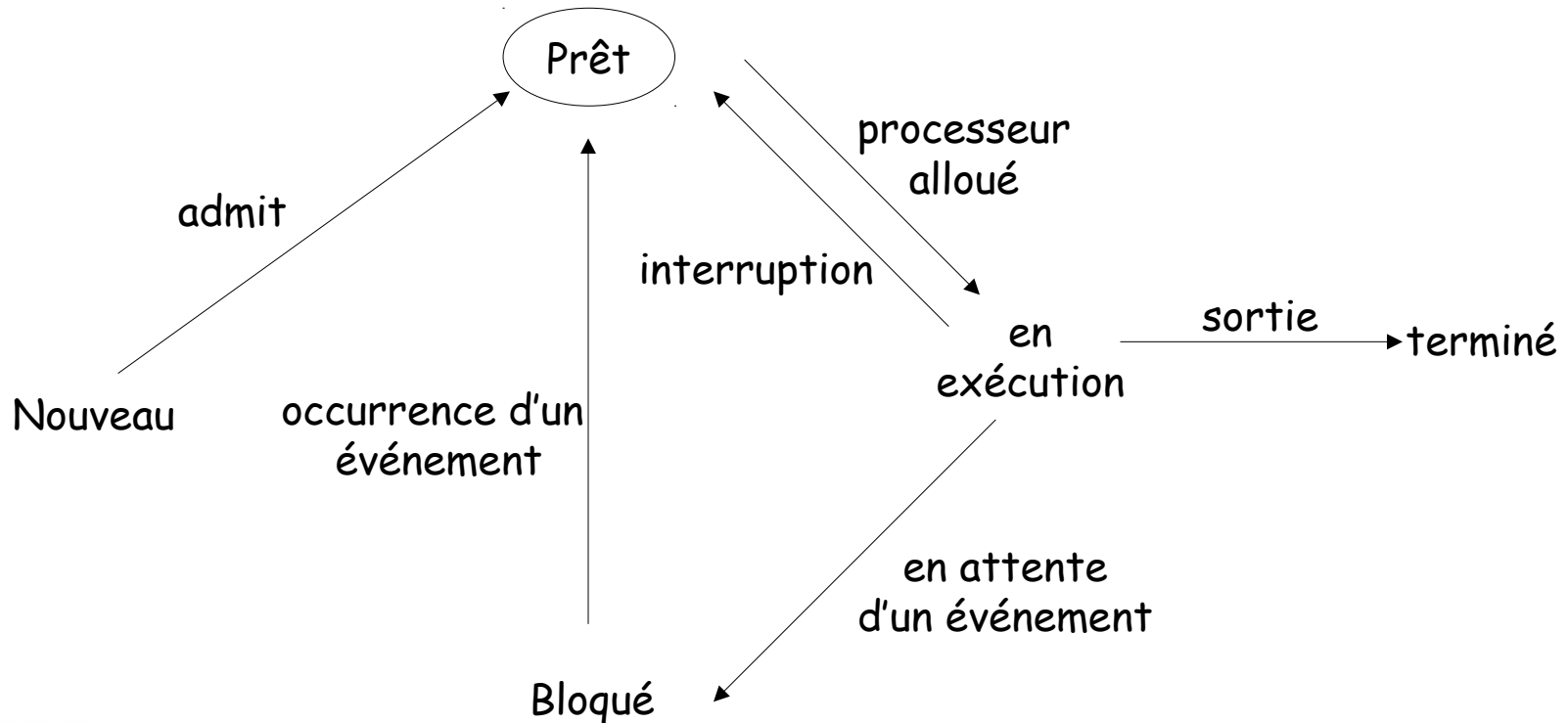
% grep -l read *

Recherche tous les fichiers contenant le mot **read** et affiche leurs noms

4. Les processus

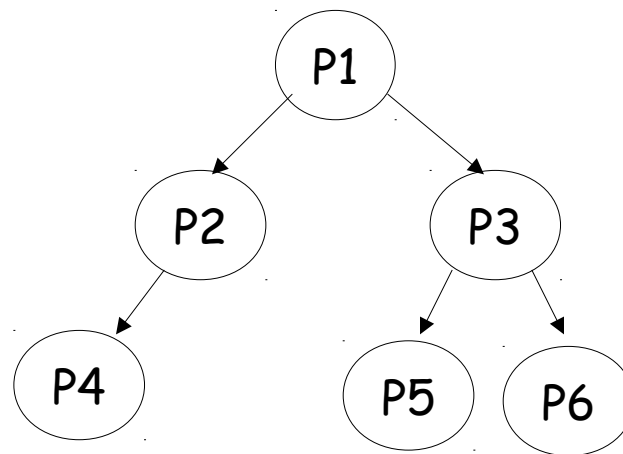
Concept de processus

- Un processus est un programme en cours d'exécution
- Le processeur traite une tâche à la fois, s'interrompt et passe à la suivante
- **Le diagramme d'état du processus**



Création d'un processus

- Les processus des utilisateurs sont lancés par un interprète de commande (shell). Ils peuvent eux même lancer ensuite d'autres processus
- Ces processus doivent ensuite pouvoir communiquer entre eux
- Le processus créateur = le **père**
- Les processus créés = les **fil**s
- Les processus peuvent se structurer sous la forme d'une arborescence



Destruction d'un processus

- 3 possibilités pour l'arrêt d'un processus
 - Normal : par lui même en ayant terminé ses opérations
 - Autorisé : par son père qui exécute une commande appropriée
 - Anormal : par le système
 - temps d'exécution dépassé
 - mémoire demandée non disponible
 - instruction invalide
 - etc.
- Le processus créateur est le seul à pouvoir exécuter l'arrêt de ses fils
- Dans plusieurs systèmes, la destruction d'un processus père entraîne la destruction de tous ses fils

Mise en œuvre

- Pour mettre en œuvre le modèle des processus, le système d'exploitation construit une table, appelé table des processus, dont chaque entrée correspond à un processus particulier
- Chaque entrée comporte des informations sur :
 - l'état du processus
 - son compteur ordinal : contient l'adresse de la prochaine instruction à extraire de la mémoire
 - son pointeur de pile : contient l'adresse courante du sommet de pile en mémoire
 - son allocation mémoire
 - l'état de ses fichiers ouverts
 - et tous ce qui peut être sauvegardé lorsqu'un processus passe de l'état élu à l'état prêt

Structure d'un processus

- L'environnement d'un processus comprend :
 - un numéro d'identification unique appelé **PID** (Process IDentifier)
 - le numéro d'identification de l'utilisateur qui a lancé ce processus, appelé **UID** (User IDentifier), et le numéro du groupe auquel appartient cet utilisateur, appelé **GID** (Group IDentifier)
 - le répertoire courant
 - les fichiers ouverts par ce processus
 - le masque de création de fichier, appelé *umask*
 - la taille maximale des fichiers que ce processus peut créer, appelé *ulimit*
 - la priorité
 - les temps d'exécution
 - le terminal de contrôle, c'est à dire le terminal à partir duquel la commande a été lancée, appelé **TTY**

Les processus : la commande ps

- Un processus est un programme qui est en cours d'exécution
- La commande **ps** donne un ensemble de renseignements sur les processus en cours d'exécution
- Syntaxe : **ps options**
- Options :
 - **-a** : affiche des renseignements sur tous les processus attachés à un terminal
 - **-l** : donne, pour chaque processus, le nom de l'utilisateur (user), le pourcentage de cpu (%cpu), la taille totale du processus dans la mémoire (size), la mémoire réservée (rss) en Ko ...
 - **-x** : affiche également des informations sur les processus non liés au terminal
 - **-w** : affiche sur 132 colonnes, utile pour voir le nom complet de la commande associée à chaque processus

% ps

PID	TTY	STAT	TIME	CMD
746	pts/3	S	00:00:00	-bash
749	pts/3	S	00:00:02	gs
848	pts/3	S	00:03:28	mozilla-bin
965	pts/3	S	00:00:00	ps

- PID : le numéro d'identification du processus
- TTY : le terminal depuis lequel le processus a été lancé
- STAT : l'état du processus au moment du lancement de la commande
 - R : le processus est en cours d'exécution
 - T : le processus est stoppé
 - S : le processus dort depuis moins de 20 secondes
 - Z : le processus en attente d'un message du noyau
- TIME : le temps d'exécution de la commande
- CMD : le libellé de la commande lancée

Arrêt d'un processus : kill

- La commande kill permet d'envoyer un signal au processus

- Syntaxes :

```
kill -signal pid  
kill -l
```

- Options :

- **-9** : demande l'arrêt du processus désigné par son **pid**
- **-l** : affiche la liste des signaux disponibles

```
% kill -l
```

```
1) HUP  2) INT  3) QUIT ...
```

```
7) EMT  8) FPE  9) KILL ...
```

```
% kill -9 1635
```

Cette commande tue le processus dont le numéro **PID** est **1635**

5. Compression et décompression

L'archivage : tar

- Cette commande permet d'archiver des fichiers ou une arborescence de fichiers, c-à-d de les regrouper dans un seul fichier

- Pour créer une nouvelle archive :

```
tar -cvf nom_archive.tar répertoire
```

- Pour afficher le contenu d'une archive :

```
tar -tvf nom_archive.tar
```

- Pour extraire les fichiers archivés :

```
tar -xvf nom_archive.tar
```

Les fichiers sont créés à partir du répertoire courant

La compression : gzip, gunzip

- **gzip** permet de compresser un ou plusieurs fichiers en remplaçant chacun par un fichier de même nom, mais avec une extension .gz

gzip options fichiers

- **Options :**

- -d : décompresse au lieu de compresser, identique à **gunzip**
- -f : force la génération du fichiers de sortie, même s'il en existe déjà un
- -r : si un des fichiers spécifiés est un répertoire, compresse son contenu récursivement
- -v : affiche le nom et la réduction de taille en pourcentage de chaque fichier

- **gunzip** décompresse les fichiers compressés dont les noms se terminent par .gz

gunzip options fichiers

```
% gzip fichier1 fichier2 fichier3
% ls
fichier1.gz fichier2.gz fichier3.gz
% gunzip fichier1.gz fichier2.gz
fichier1 fichier2 fichier3.gz
```

6. Utilisation du réseau

Les commandes : who et finger

- **who** liste les noms des utilisateurs actuellement connectés, leur terminal, la machine utilisée et l'heure à laquelle ils se sont connectés

```
% who
```

and7483	ttyp0	cis450	Oct 28	09:38
del4820	ttyp1	cis324	Oct 28	09:50
elkh3481	ttyp2	cis128	Oct 28	10:02

- **finger** affiche des données sur un ou plusieurs utilisateurs
finger options utilisateurs

Chaque utilisateur peut être spécifié par son login, prénom ou nom de famille

- **Options :**
 - -l : format long
 - -s : format court

Le réseau

- Les réseaux relient les ordinateurs pour que différents systèmes partagent rapidement et facilement des informations
- Le réseau permet le partage des ressources : matérielles, logicielles, données
 - imprimantes, espace disque, périphériques coûteux
- Les systèmes Linux permettent de tester quels systèmes sont lancés, d'accéder aux fichiers présents sur les systèmes distants, de communiquer par courrier électronique, etc.
- Plusieurs applications réseau :
 - transfert de fichiers : ftp, tftp, rcp, scp
 - connexion sur un ordinateur distant : telnet, rlogin, rsh, ssh
 - courrier électronique : mail, talk
 - etc.

Transfert de fichiers : File Transfer Protocol

- Cette commande permet de transférer des fichiers vers et à partir d'une machine distante

ftp **nom_machine**

- Le champ **nom_machine** est une adresse Internet :
 - absolu **131.184.96.542**
 - symbolique **cis149.univ-bordeaux.fr**

```
% ftp cis149.univ-bordeaux.fr
Connected to 131.184.96.542
Name: votre_login
Passwd:
Connected.
ftp>
```

- Le prompt **ftp>** invite l'utilisateur à saisir une commande : les plus utilisées sont : **cd**, **lcd**, **put**, **get**, **binary** et **help**

Connexion sur une machine distante : TERminal NETwork protocol

- La connexion se fait par la commande : **telnet nom_machine**
- Lorsque la connexion est établie, l'utilisateur s'identifie sur la machine distante et entre en session sur cette machine

```
% telnet cis149.univ-bordeaux.fr
Trying ...
Connected to cis149.univ-bordeaux.fr
Name: votre_login
Passwd:
Last login: wed Mar 3 16 ...
<cis149> % logout
Goodby ...
Connection closed.
%
```

- Si votre machine est connecté à Internet, il est très imprudent d'utiliser **telnet** : le mot de passe est transféré en clair sur le réseau
- Il est donc préférable d'utiliser **ssh**

7. Les éditeurs de texte

L'éditeur vi

- vi est un éditeur entièrement en mode texte : chacune des commandes se fait à l'aide de commandes texte
- vi est peu pratique, très puissant, très utile en cas de non fonctionnement de l'interface graphique
- Quand vi devient actif :
 - un ~ apparaît à gauche de chaque ligne de l'écran,
 - vi est alors en **mode commande** et attend votre première instruction
- vi possède 2 modes :
 - **mode commande** : permet de taper des commandes
 - **mode insertion** : permet de saisir du texte en ajoutant du texte après ou avant le curseur
- Pour passer du **mode commande** en **mode insertion**, tapez :
 - **a** pour insérer du texte **après** le curseur
 - **i** pour insérer du texte **avant** le curseur

Créer un fichier vi

- Lancer vi en tapant `vi`
- Un écran comportant une colonne remplie de tildes s'affiche
- Passer du mode commande en mode insertion en appuyant sur la touche `a` (n'appuyer pas sur Entrée)
- Vous pouvez insérer des caractères sur la première ligne. Le caractère `a` n'apparaîtra pas à l'écran
- Ajouter des lignes de texte, vous pouvez utiliser la touche Correction pour supprimer les erreurs de la ligne en cours
- Passer du mode insertion en mode commande en appuyant sur la touche Echap
- Enregistrer en tapant : `:w nom_du_fichier`
- La ligne d'état confirme cet action en affichant :
 `"nom_du_fichier" [New File] 4 lines, 46 characters`
- Quitter vi en tapant : `:q`

Quelques commandes : vi

- **Commandes de base :**

:q	quitte l'éditeur
:q!	force l'éditeur à quitter
:wq	sauvegarde le document et quitte l'éditeur
:nom_du_fichier	sauvegarde le document sous le nom nom_du_fichier

- **Commandes d'édition :**

x	efface le caractère actuellement sous le curseur
dd	efface la ligne actuellement sous le curseur
dxd	efface x lignes à partir de celle actuellement sous le curseur
nx	efface n caractères à partir du caractère actuellement sous le curseur
r	remplace le caractère actuellement sous le curseur
cw	modifie le mot courant à partir de la position du curseur
cc	modifie la ligne entière