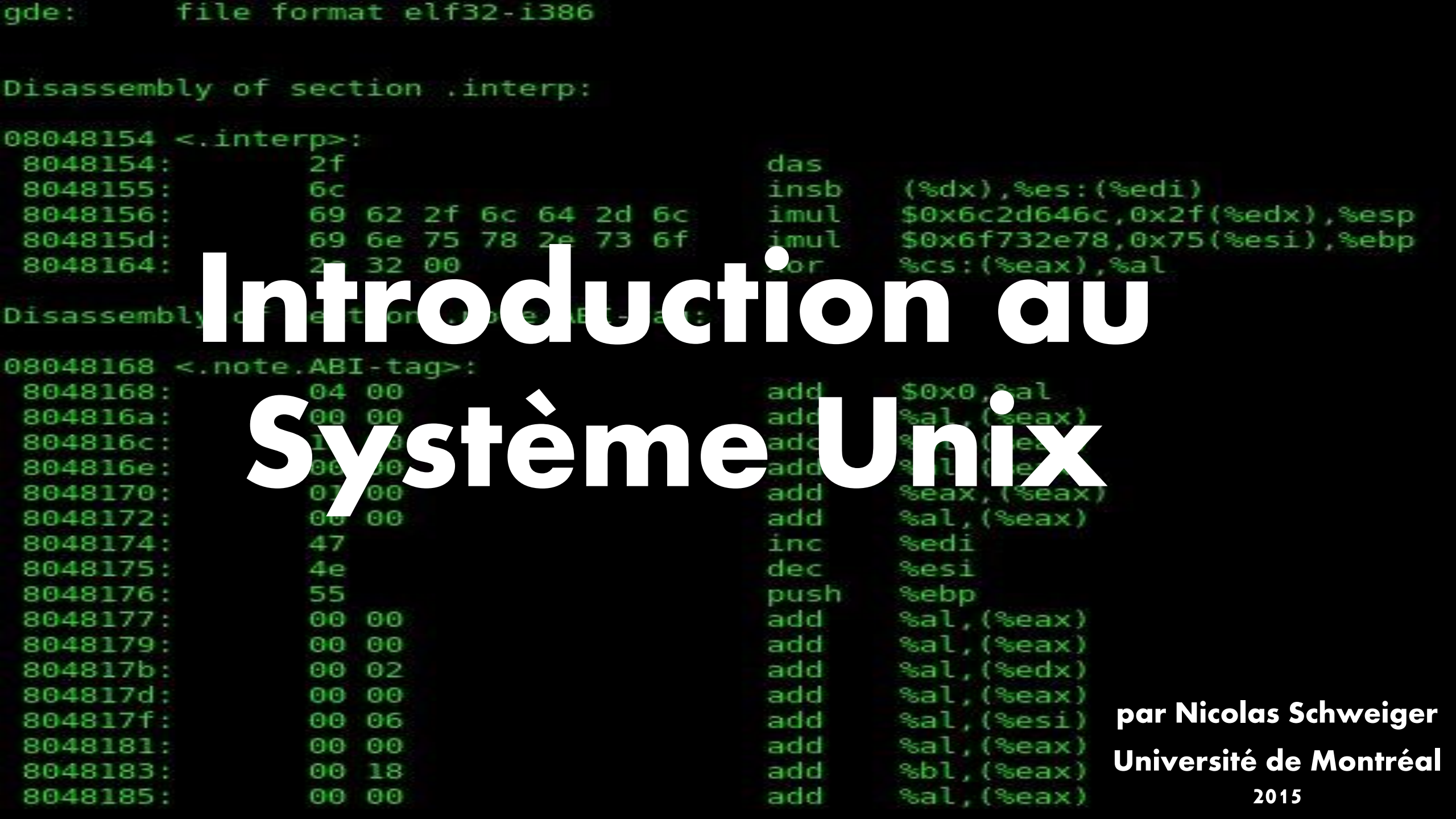

Introduction au Système Unix

par Nicolas Schweiger
Université de Montréal

2015

Les systèmes d'exploitation UNIX populaires

- Linux
- Mac OSX
- FreeBSD
- NetBSD
- OpenBSD
- Oracle Solaris
- IBM AIX
- Hewlett-Packard HP-UX

La hiérarchie d'un système informatique

Les applications (programmes)
Interpréteur de commande et interface graphique
La gestion de la mémoire
Le système d'exploitation (le noyau)
Le matériel physique (la machine)

Histoire de UNIX

- 1960 MULTICS par Bell Labs, GE et le MIT, projet abandonné
- 1966 Bell Labs (AT&T) débute un SE pour leur usage interne, développé par Ken Thompson et Dennis Ritchie. Ritchie inventera également le language C, successeur du language B de Thompson.
- 1969 Première version d'UNIX, le nom vient de MULTICS
- 1973 Réécriture complète d'UNIX en language C
- 1974 AT&T donne des licences d'utilisation de leur UNIX aux universités, donnant naissance ainsi à BSD (Berkeley Software Distribution) grâce au travail de recherche du CSRG (Computer Systems Research Group) de l'Université de Berkeley.
- 1978-80 AT&T présente et autorise le clônage de leur UNIX par d'autres constructeurs (Ultrix DEC, BSD Sun, AIX IBM, AUX Apple, XENIX Microsoft, Irix SGI)
- 1991 La première version de Linux qui est un clône du système MINIX

Histoire de UNIX

- En 1984, au MIT, Richard Stallman débute le mouvement Open source (logiciel libre) avec la création de la Free Software Foundation (FSF). La naissance de GNU est donc de mise, qui est un acronyme pour GNU's not UNIX, le nom UNIX étant une marque de commerce de AT&T, donc inutilisable. Le projet GNU actuel fournit l'ensemble des utilitaires systèmes présents dans une distribution Linux, le noyau de Linux dit le Linux kernel est téléchargeable séparément sur www.kernel.org. Le kernel de linux entouré de ces outils forme la distribution Linux. Ex: Debian Linux, Red Hat Linux, Ubuntu Linux, etc.

- UNIX (ses dérivés) est le système d'exploitation le plus populaire au monde. Il supporte un très grand nombre d'architecture dû à sa portabilité grâce au language C. Il est multi-utilisateur et multi-tâche. Il roule également sur la majorité des téléphones et tablettes portables: Iphone(IOS), Android (Linux), Blackberry (QNX).

Mécanisme d'un système UNIX

- Temps partagé (Time sharing) le système attribue à chaque application un temps d'accès aux ressources du système.
- Préemption (Preemption) le système juge lorsqu'il est nécessaire d'interrompre la priorité d'une application sur les ressources du système (cpu, mémoire, par exemple) pour donner priorité à une autre application.
- Réentrance (Reentrancy) cela permet un système multi-utilisateur efficace, la même application qui est utilisée par plusieurs utilisateurs différents ne sera chargée en mémoire qu'une seule fois, si toutefois le language de programmation de cette application le permet.

Mécanisme d’un système UNIX

- Gestion des ressources UNIX gère les ressources de l’ordinateur, le noyau (kernel) répartit ses ressources de façon transparente. L’accès aux divers fonctions du noyau est possible pour l’utilisateur via les appels système (system calls) invoqués dans le code d’une application.
- Gestion des disques le système gère la maintenance, l’accès et l’organisation aux divers unités de stockage du système (disques durs).
- Communication et développement UNIX offre un système de communication entre les utilisateurs, ainsi qu’un environnement de programmation complet avec compilateur de code, éditeur texte et de nombreux outils de programmation.
- Interprèteur de commande l’utilitaire le plus important pour les utilisateurs d’un système UNIX est l’interprèteur de commande qu’on appele un Terminal (Shell).

Le Shell UNIX

Il existe plusieurs types de shell différents sous UNIX, chacun possède son language de script particulier, ainsi que ses propres outils, en voici quelques-uns:

- Thompson Shell premier shell unix, inventé par Ken Thompson en 1971 chez Bell Labs
- SH (Bourne shell) inventé par Steve Bourne chez Bell Labs en 1977, il est destiné à remplacer le Thompson shell. Il offre un language pour produire des scripts.
- BASH (Bourne Again Shell) inventé par Brian Fox en 1989 pour le projet GNU, il est une variante du shell SH. Il est le plus populaire et celui par défaut sur Linux et Mac OSX.
- ASH (Almquist Shell) c'est le remplacement du shell SH original pour les systèmes FreeBSD et NetBSD.
- KSH (KornShell) il est inventé par David Korn chez Bell Labs en 1983, son language de script s'apparente au C et il est le shell par défaut sur OpenBSD (fork).

Mode graphique sous UNIX

Le mode graphique sous un système UNIX fonctionne à partir d'un serveur graphique. Le serveur graphique standard UNIX est X Window, qui était autrefois X11, il est maintenant Xorg (il sera bientôt remplacé par Wayland). Il est le serveur graphique par défaut sur tous les sytèmes UNIX, mis à part Apple, qui utilise Quartz Compositor. Xorg est toutefois disponible sur Mac OSX via XQuartz qui est l'implantation du protocol X11 avec une série de patch d'Apple, ce qui permet de pouvoir rouler n'importe quelle application UNIX sur Mac OSX.

Une interface graphique utilisateur (GUI) est cependant nécessaire pour qu'un utilisateur puisse obtenir un environnement de travail graphique roulant sur le serveur graphique, voici les plus populaires sous UNIX :

- CDE première version en 1993, c'est une interface graphique développée en partenariat avec HP, IBM, Sun Microsystem et AT&T (Unix System Laboratories). Elle était installée sur la plupart des UNIX commerciaux de l'époque.
- KDE première version en 1998, c'est une des interfaces les plus populaires.
- GNOME première version en 1999, son clône appelé Unity est l'interface graphique par défaut sous Ubuntu Linux.
- Aqua première version en 2000, c'est l'interface graphique standard de Mac OSX.

Les fichiers sous UNIX

Sous un système UNIX, tout ce qui se trouve sur une unité de stockage (disque dur du système, tape backup, par exemple) est considéré comme un fichier. Il existe six types de fichiers principaux composant l'arborescence d'un système UNIX (on peut voir de quel type de fichier il s'agit avec la commande ls -l, le premier caractère qui apparaît désigne le type).

- Les répertoires (d) ce sont des fichiers regroupant des informations sur les autres fichiers.
- Normaux (-) ce sont des fichiers regroupant les informations des utilisateurs en format texte ou binaire (exécutable).
- Lien symbolique (l) ce sont des fichiers qui pointent tout simplement vers d'autres fichiers.
- Tube nommé (named pipe) (p) ce sont des fichiers permettant la communication entre les processus, cela sert généralement à rediriger la sortie d'un processus vers l'entrée d'un autre (commande mkfifo).

Les fichiers sous UNIX

- Socket (s) ce sont des fichiers utilisés pour la communication inter-processus, ils permettent la communication entre deux processus du système (échange de données). Un fichier socket peut être utilisé pour se connecter par exemple à une base de données MySQL sans avoir à utiliser le protocol TCP/IP (réseau).

- Spéciaux (device) (c) et (b) ces fichiers permettent l'accès aux périphérique de l'ordinateur (disques, écran, imprimante, clavier, souris, carte résea, carte graphique, etc.).

Les caractéristiques principales d'un fichier sous UNIX est le nom de celui-ci, la date et heure de création, la date de modification, la taille, les permissions d'accès, ainsi que le propriétaire.

Les systèmes de fichiers sous UNIX

Le système de fichier est la partie la plus importante d'un système d'exploitation; c'est l'endroit où toutes les applications enregistrent et retrouvent les informations du système.

Le système de fichier assure la conservation des données et réalise les fonctions d'accès à celles-ci. Il existe une multitude de système de fichiers différents, voici les principaux :

- EXT4 Linux
- HFS+ Mac OSX
- UFS FreeBSD
- ZFS Oracle Solaris, FreeBSD
- NTFS Windows XP/7/8/Server 2003/2008/2012
- FAT32 Windows XP/7/8 utilise surtout pour les clés USB

Structure des fichiers sous UNIX

Rappel sur les unités de mesure de stockage d'un ordinateur

1 bits = valeur de 0 ou 1

1 octet = 8 bits, séquence de huit 0 ou 1, ex: 1010 1111

1 ko = 1024 octets

1 mo = 1024 ko

1 go = 1024 mo

1 to = 1024 go

Note sur le nom des fichiers stockés sur une unité de stockage

L'ensemble des systèmes UNIX sont dit case sensitive, une minuscule est différente à une majuscule. Cependant, Mac OSX vient par défaut case insensitive, un caractère minuscule est equivalent à une majuscule, chose très inhabituelle pour un système UNIX. C'est aussi le cas pour le système Windows qui est également case insensitive, mais il existe des solutions pour les rendre case sensitive (OS X et Windows).

Structure des fichiers sous UNIX

La structure des fichiers sous un système UNIX est dite à structure d'arbre. L'arbre est constitué d'un répertoire racine unique appelé le root directory qui est représenté par le caractère ***/***. Chaque fichier dans le système possède un chemin unique à partir de la racine. Contrairement au système windows où chaque disque possède sa lettre assignée, ainsi que sa propre structure indépendante (C:, D:, E:, etc.), tous les disques et autres périphériques sous un système UNIX possèdent un chemin unique à partir du répertoire racine.

Il existe deux méthodes pour accèder à un fichier dans un système de fichier dit à structure d'arbre :

- Nom absolu /home/tata/toto.txt
- Nom relatif ~/toto.txt, ~/ désigne le répertoire appartenant à l'utilisateur tata (/home/tata dans ce cas ci). Si on se situe dans /home/tata/titi, un ls -l ../toto.txt fonctionne également.

Structure des fichiers sous UNIX

Arborescence générale d'un système UNIX

/	la racine
/etc	les fichiers de configuration, les scripts de démarrage, les crons
/dev	les fichiers des périphériques
/bin	les commandes importantes
/sbin	les commandes cruciales
/usr	les installations générales
/home	les répertoires et fichiers des utilisateurs
/root	le dossier de l'administrateur
/mnt	les disques du système
/opt	les installations optionnelles
/var	les logs du système, les bases de données, les données cache du système, les courriels du mail server
/tmp	les fichiers temporaires créés par les applications du système

Permission des fichiers sous UNIX

Tous les fichiers sous un système UNIX disposent de permission d'accès. Chaque utilisateur possède son user id (UID) et son groupe id (GID). Voir la commande id pour obtenir votre uid et gid.

Chaque fichier possède des attributs indiquant à quel propriétaire il appartient. Il existe trois niveaux de confidentialité sur UNIX :

- Le propriétaire
- Le groupe
- Les autres

Il y a 9 caractères désignant les droits d'accès sur un fichier

On peut voir ces caractères sur un fichier avec la commande ls -l nomdufichier :

-rwxr-x--x 1 nick langlab 135325 Aug 20 13:11 gde

ici, le fichier gde est un fichier normal, le premier caractère de la séquence étant un -

le propriétaire du fichier est l'utilisateur nick, il appartient également au groupe langlab

la première séquence rwx appartient au propriétaire nick, la deuxième séquence r-x au groupe langlab et la dernière séquence --x à tout autre utilisateur ou groupe sur le système. On voit aussi sa taille, ainsi que sa date de dernière modification.

Permission des fichiers sous UNIX

Le codage des permissions sous UNIX

r signifie read, w signifie write, x signifie execute

chiffre 7 désigne rwx

chiffre 6 désigne rw-

chiffre 5 désigne r-x

chiffre 4 désigne r--

chiffre 3 désigne -wx

chiffre 2 désigne -w-

chiffre 1 désigne --x

chiffre 0 désigne ---

dans l'exemple précédent, rwx r-x --x en binaire :

111 101 001

cela donne 751 en décimal, on utiliserait la commande chmod 751 nomdufichier pour placer une permission de ce type sur un fichier donné. Donc, dans ce cas-ci, l'utilisateur nick aurait droit d'écriture, de lecture et d'exécution, le groupe langlab de lecture et d'exécution, et tous les autres seulement d'exécution.

Opérations sur les fichiers UNIX

Les opérations principales sur les fichiers normaux sous UNIX avec quelques commandes de base :

touch nomdufichier création d'un nouveau fichier vide.

file nomdufichier determine le type de fichier.

ls –l nomdufichier affiche les caractéristiques du fichier

cat ou less nomdufichier affichage du contenu du fichier, attention à utiliser seulement sur les fichiers textes non binaire (exécutable).

cp nomdufichier destination copier le fichier

mv nomdufichier destination déplace le fichier

rm nomdufichier supprime le fichier

chmod 755 nomdufichier change la permission du fichier pour rwxr-xr-x

chown utilisateur:groupe nomdufichier change le propriétaire et le groupe du fichier

find ~/ -name nomdufichier cherche le fichier dans le système à partir de votre dossier utilisateur

vim nomdufichier editeur texte en terminal

kwrite nomdufichier editeur texte en mode graphique (KDE)

Opérations sur les fichiers UNIX

mkdir nomdudossier crée un répertoire

rmdir nomdudossier supprime un répertoire, il doit être vide avec cette commande

mv nom nouveaunom renomme un répertoire ou un fichier

cd nomdudossier déplace à l'intérieur d'un répertoire

ls –la * affiche le contenu du répertoire actuel

cp –r nomdudossier destination copie le répertoire

pwd affiche le répertoire de travail actuel

man commande montre le manuel de la commande

history affiche l'historique des commandes

grep "pattern" nomdufichier cherche une chaîne de caractères spécifiques dans un fichier texte

head fichier montre les 10 premières lignes du fichier

tail fichier montre les 10 dernières lignes du fichier

Connexion à un système UNIX

- L'accès à un système UNIX peut s'effectuer de deux façons soit physiquement sur la machine en question ou à distance à partir d'un autre poste de travail. Dans les deux cas, il est nécessaire de posséder un nom d'utilisateur personnel, ainsi qu'un mot de passe.

- L'accès à distance vers une machine UNIX se fait de nos jours via le protocol SSH, successeur du protocol non-sécuritaire Telnet d'autrefois. Le protocol SSH offre une connexion sécuritaire et encrypté à la machine UNIX distante. La façon générale de procéder est d'ouvrir un terminal local sur votre machine et d'utiliser la commande ssh. Voici un exemple de connexion de l'utilisateur titi à la machine distante toto (il est nécessaire de connaître le nom d'hôte de la machine distante) :

```
ssh titi@toto
```

- Une fois connecté à la machine distante UNIX un message d'accueil s'affiche et le terminal distant est prêt à recevoir vos commandes.

Les processus sous UNIX

Un processus peut être défini comme étant un programme en cours d'exécution sur le système. Un système d'exploitation comporte plusieurs dizaines, voir centaines et même milliers de processus roulant simultanément. C'est le système d'exploitation lui-même qui gère ces processus et leur attribue les ressources nécessaires à leur fonctionnement. Le processeur de l'ordinateur est l'unité centrale sur laquelle le processus est executé. Plus un ordinateur comporte de processeur (système smp, multi cores/threads), plus il sera efficace à gérer un grand nombre de processus roulant simultanément.

Un processus peut avoir des flots d'éxécution multiples (multithreaded), dans ce cas, un processus peut possèder la capacité d'utiliser plusieurs processeurs (cores,threads) simultanément, accélérant ainsi la vitesse d'exécution de celui-ci (processus de jeu, de bioinformatique, par exemple).

Le processeur d'un ordinateur est capable d'arrêter un processus pour prioriser un autre processus plus important, et puis par la suite, reprendre le processus arrêté et en continuer le traitement.

Les processus sous UNIX

Voici quelques commandes de base pour la gestion des processus sous UNIX:

top ou htop visualisation en temps réel de l'ensemble des processus en cours d'exécution sur le système.

ps ux affiche les processus en cours sur le système propre à votre utilisateur.

nomprogramme & lancer votre programme en background.

nice –n +19 nomprogramme & lancer votre programme en priorité d'exécution minimale.

nice –n 20 nomprogramme & lancer votre programme en priorité d'exécution maximale.

renice +19 #pidprocess mettre votre processus en priorité d'exécution minimale.

renice -20 #pidprocess mettre votre processus en priorité d'exécution maximale.

kill -9 #pidprocess tuer votre processus.

kill –STOP #pidprocess mettre en pause votre processus.

kill –CONT #pidprocess reprendre l'exécution du processus en pause.

gde: file format elf32-i386

Disassembly of section .interp:

UNIX Timeline