Download as PDF

TP n°1: Prise en main d'Unix

Objectifs:

- Découvrir son environnement de travail
- Commencer à manipuler son environnement de travail
- Être capable de taper ses premières commandes

Éléments techniques abordés:

- *Les commandes*
  - `exit`, `logout`, `passwd`
  - `pwd`, `cd` , `grep`, `wc`
  - `alias`, `who`, `w`, `ls`
  - `man`, `finger`, `id`
  - `ps`, `top`, `kill`
- *Les répertoires*
  - `/home`, `/users`
- *Les fichiers*
  - `/etc/passwd`, `/etc/group`

La première session

La connexion

La connexion sur un système UNIX [https://fr.wikipedia.org/wiki/Unix] utilise un système d'authentification basé sur un identifiant (login) et un mot de passe. Le système demande tout d'abord le login — c'est la phase d'identification —, puis le mot de passe — c'est la phase d'authentification. Cette méthode est très répandue et les systèmes sans authentification (comme DOS [https://fr.wikipedia.org/wiki/Disk_operating_system], Windows 95 [https://fr.wikipedia.org/wiki/Windows_95] ou 98 [https://fr.wikipedia.org/wiki/Windows_98]) ont quasiment disparu. Si la méthode est devenue courante, les contraintes et les règles de sécurité qui y sont rattachées sont souvent méconnues des utilisateurs. L'identifiant peut être local (connu uniquement sur la machine utilisée) ou réseau (dans ce cas les identifiants peuvent être stockés dans un annuaire LDAP [https://fr.wikipedia.org/wiki/Lightweight_Directory_Access_Protocol])

L'identification

Le nom de login. Le nom de login permet à l'utilisateur de s'identifier. Lors de l'identification, le système recherche, parmi la liste des utilisateurs, le nom saisi. Ce nom peut être choisi librement (sauf politique particulière imposée par l'administrateur et problème d'homonymie), mais il est soumis à certaines contraintes. Dans votre cas précis votre nom de login est est construit à partir de votre numéro d'étudiant.

Les systèmes UNIX différencient majuscules et minuscules.

En effet, les systèmes UNIX utilisent la table ASCII (*American Standard Code for Information and*

Interchange) qui distingue majuscules et minuscules. Ainsi, une saisie du nom de login en majuscules au lieu de minuscules provoque une tentative d'identification sous un « autre nom ».

L'authentification

Le mot de passe. Le rôle du mot de passe est de confirmer que l'utilisateur qui saisit le nom de login est bien le propriétaire du compte. Ce mot de passe est secret. Lors de la frappe, l'écho de chaque caractère est remplacé par des « * », ou bien n'apparaît pas du tout. Cela évite de fournir aux regards indiscrets une information sur la longueur du mot de passes.

La distinction entre majuscules et minuscules vaut également pour le mot de passe: si le système est en majuscules (par exemple : lorsque la touche CAPS LOCK est activée) lors de la saisie du mot de passe, alors le mot de passe ne sera pas reconnu lors de l'authentification et un message d'erreur apparaitra.

La session

Une fois qu'il est authentifié par le système, l'utilisateur se retrouve connecté: une session est ouverte. L'invite de saisie de l'identification change selon l'interface, qui peut être graphique ou textuelle.

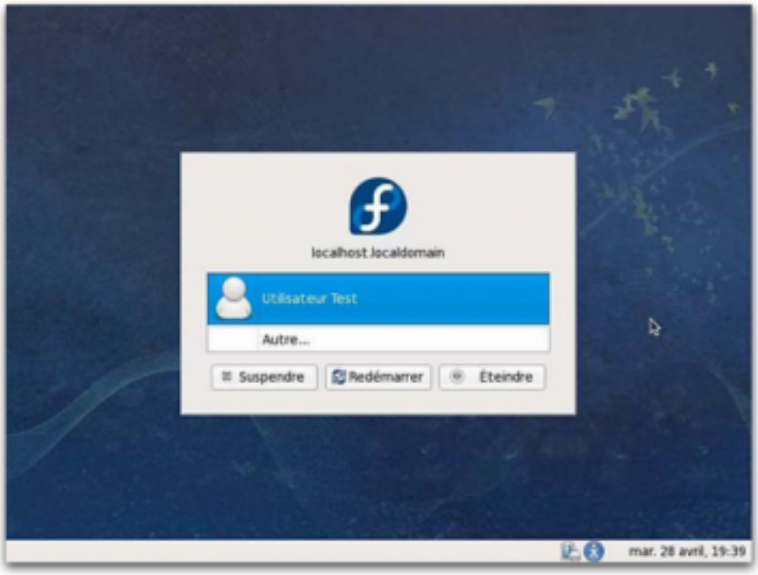
Les interfaces

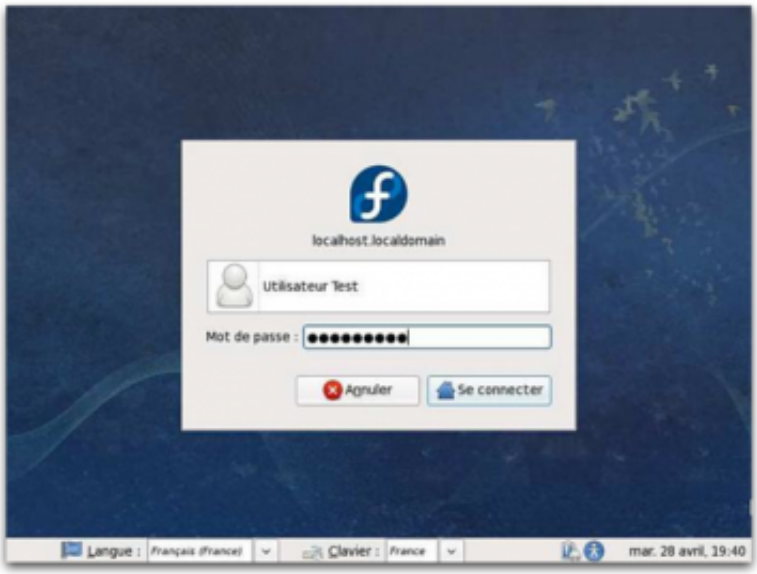
Il existe deux interfaces, ou modes connexions:

1. l'interface graphique, également appelée *mode graphique*;
2. l'interface texte, également appelée *mode commande*.

Les deux interfaces sont présentes dans l'environnement UNIX particulier aux « gros ordinateurs », et dans l'environnement Linux dédié au micro-ordinateurs. Mais s'il est plus courant de trouver l'interface texte sur les serveurs administrés sous UNIX et l'interface graphique pour Linux vous devez être capable de manipuler les deux.

Remarque: Sous Linux, on peut facilement basculer du mode graphique vers l'interface texte avec l'association de touches CTRL+ALT+F1 (touche contrôle, touche ALT et touche de fonction F1), puis revenir en mode graphique avec CTRL+ALT+F7. Chaque combinaison de touches CTRL+ALT+F1, CTRL+ALT+F2, etc. ouvre une nouvelle console « virtuelle » en mode texte. Seule l'association CTRL+ALT+F7 bascule vers une console graphique.





Le mode graphique. Le mode graphique repose sur le protocole X-Window [https://fr.wikipedia.org/wiki/X-Window], ou X11 [https://fr.wikipedia.org/wiki/X-Window]. Ce protocole est employé lors d'une ouverture de session à partir d'un terminal. Un terminal est un programme qui permet de se connecter à une machine. Les serveurs UNIX proposent généralement un bureau graphique s'appuyant sur X-Window: il s'agit de CDE [https://fr.wikipedia.org/wiki/Common_Desktop_Environment] (*Common Desktop Environnement*). Ce bureau normalisé présente la même interface sur l'ensemble des serveurs qui l'implémentent. Il propose des fonctionnalités et des utilitaires intégrés dans une barre de menus, ainsi qu'un bureau virtuel, disponible sous la forme de quatre écrans différents, accessibles d'un simple clic dans la barre. Vous pouvez modifier cette interface afin de l'adapter à vos besoins (ajout d'outils dans la barre de menus, choix du nombre de bureaux virtuels, choix de la police, etc.).

Linux dispose de ses propres bureaux graphiques. Les deux principaux se nomment KDE [https://fr.wikipedia.org/wiki/Kde] (*Kool Destop Environnement*), GNOME [https://fr.wikipedia.org/wiki/GNOME] (*Gnu Network Object Model Environnement*) et Xfce [https://fr.wikipedia.org/wiki/Xfce] fondé sur trois principes : rapidité, économie de ressources et simplicité d'utilisation. L'environnement utilisé en TP est Xfce. KDE est une adaptation de CDE à l'environnement Linux. Il est très utilisé et présente le double

avantage d'être très proche de l'ergonomie de CDE et du bureau Windows. Le fait de choisir KDE ou GNOME ou Xfce n'a aucune incidence sur le fonctionnement des applications, qui sont en majorité indépendantes du type de bureau. Ce choix est principalement dicté par l'appréciation de la présentation graphique et des outils associés à chaque interface. Que l'on se serve de l'un ou de l'autre bureau, une bannière graphique nous invite à saisir nom de login et mot de passe.

Le mode commande. L'interface texte est plus simple, et souvent privilégiée par les administrateurs et les développeurs. Une fois que l'on est connecté, le bureau s'affiche à l'écran. Il suffit alors d'ouvrir une fenêtre de terminal pour demander au système de réaliser des commande. Pour ouvrir un terminal dans un environnement graphique :

1. Ouvrez le menu Applications. Il est représenté par une “souris” (à droite en haut ou bien en bas de votre écran). Le menu peut quelquefois aussi être représenté par un petit “pied” voire un petit manchot.
2. Dans le sous-menu Outils système, cliquez sur Terminal. Dans certaines versions de Fedora, vous le trouverez dans le sous-menu Accessoires au lieu du sous-menu Outils système.

Mise en pratique

Pour accéder aux machines de l'UFR de Maths-Info vous devrez toujours utiliser votre identifiant (login) et votre mot de passe. Pour les primo-arrivants (première inscription à l'Université Paris Descartes), il vous faut votre carte étudiant. Votre login et votre mot de passe étudiant sont construits de la manière suivante :

- Les 5 derniers chiffres du numéro d'étudiant servent à construire le login précédé de “**ii**” pour les primo-arrivants 2016.
- “**tp**” en minuscules suivi des 5 derniers chiffres de votre numéro étudiant.

Première commande

Lorsqu'une session est ouverte, vous pouvez vérifier qui a ouvert cette session en utilisant la commande whoami (Qui suis-je ? en anglais).

```
$ whoami
dupont
```

Le changement du mot de passe

Si le mot de passe a été fourni par l'administrateur (ce qui est votre cas), il est recommandé de le changer dès sa première connexion (ce qui normalement est également votre cas).

Vous êtes légalement responsables des ressources informatiques qui vous sont confiées.
Lors de votre inscription vous avez signé la charte des utilisateurs des moyens informatiques de l'université Paris Descartes [https://app.parisdescartes.fr/images/esup/Charte_Universite_Paris_Descartes.pdf]

Un bon mot de passe doit mélanger des lettres (minuscules et majuscules), des chiffres et au moins un caractère de ponctuation. Il ne doit pas non plus être évident: on évitera son prénom, le nom de son chien, etc. ! Enfin, plus le mot de passe est long, plus il sera difficile à pirater. La plupart des systèmes UNIX, proposent l'activation d'une librairie, la CrackLib. Au moment du changement du mot de passe, elle permet de vérifier plusieurs critères (paramétrables) afin d'éviter de choisir un mot de passe trop simple: présent dans un dictionnaire, trop proche du nom, du prénom, de l'ancien mot de passe (si c'est un changement), trop court… Si le système est activé, vous éviterez un certain nombre de problème.

Astuce: Ne choisissez pas un mot de passe trop banal et faites attention aux majuscules et aux minuscules

lors de la saisie. Éviter d'utiliser le pavé numérique, qui parfois n'est pas actif sur le poste de travail lors de cette phase d'authentification. Il est difficile de détecter que le pavé ne génère aucun numéro quand l'écho de la frappe n'est pas visible ! Enfin, créez un mot de passe d'au moins 8 caractères.

La commande passwd permet de changer et donc de choisir son mot de passe. Cette commande demande d'abord d'entrer le mot de passe actuel, afin d'éviter qu'une tierce personne ne puisse changer le mot de passe d'un utilisateur qui serait parti en laissant sa session ouverte. Elle demande ensuite de saisir deux fois le nouveau mot de passe, afin d'éviter cette fois des erreurs, qui seraient ici catastrophiques (ne voyant pas l'écho de sa frappe, l'utilisateur ne peut détecter l'erreur de frappe).

```
$ passwd
Enter login(LDAP) password:
New password:
Re-enter new password:
LDAP password information changed for ifXXXXX
passwd : le mot de passe a ete mis a jour avec succes
```

Attention: La commande de changement de mot de passe s'écrit passwd mais il est courant, dans le monde UNIX, de la nommée « password » (mot de passe en anglais). La prononciation diffère de l'écriture.

La sécurité de vos comptes

Chaque année de nombreux sites sont piratées et souvent les données sont alors publiées sur internet. Dans l'historique du compte twitter associé à ce cours [https://twitter.com/updcbi] vous trouverez de nombreux exemples. Les pirates (les mêmes ou d'autres) utilisent ces données pour réaliser d'autres piratages en utilisant les informations mail ou login et mot de passe sur d'autres sites. Sur des millions d'utilisateurs, cela conduit à un nombre non négligeable de succès. Le site https://haveibeenpwned.com/ [https://haveibeenpwned.com/] vous permet de vérifier si votre adresse mail figurait dans l'une de ces bases de données qui ont été piratées et pour lesquelles les données ont été publiées sur internet (gratuitement). S'il y a une correspondance pour l'une de vos adresses mail alors il est **crucial** de ne plus jamais utiliser le mot de passe qui y était associé sur l'ensemble de vos comptes

Pour vous faire une idée des mots de passe à éviter voici un premier lien à explorer : Découvrez les pires mots de passe des membres du site Ashley Madison [https://www.01net.com/actualites/decouvrez-les-pires-mots-de-passe-des-membres-du-site-ashley-madison-914805.html]

La déconnexion

La déconnexion implique l'arrêt de la session en cours. Selon l'interface de connexion (graphique ou texte), la méthode pour se déconnecter change. Parfois une simple commande suffit, alors qu'en mode graphique il faut arrêter l'ensemble des logiciels qui contrôle la session.

En mode graphique

Pour terminer la session, il vous suffit de:

1. Cliquer sur le menu Système en haut à gauche puis de
2. Choisir déconnexion

Attention: Il est fortement conseillé d'attendre l'affichage de la bannière de login (écran où vous entrez votre identifiant et votre mot de passe) afin d'être sûr que la phase de déconnexion s'est bien déroulée correctement et complètement.

En mode commande

En mode commande il suffit de taper la commande `logout`. Cette commande aura pour effet de terminer votre session. Il vous sera alors proposer de vous connecter à nouveau. Il également possible d'utiliser la commande `exit` qui permet de terminer le shell courant. Finalement, la combinaison CTRL+D (touche « contrôle » et « d » enfoncées simultanément) provoque le même résultat.

Attention. L'arrêt du shell ou la déconnexion arrête tous les logiciels que vous auriez préalablement lancés.

Exercice 1.1 (Connexion et déconnexion)

1. Connectez vous via l'interface graphique. Le mot de passe par défaut qui vous a été attribué est la chaîne “tp” suivit des 5 derniers chiffres de votre carte d'étudiant.
2. Lancer un terminal.
3. Vérifiez votre identifiant en utilisant la commande `whoami`
4. Changez votre mot de passe en utilisant la commande `passwd`. Pour vous aider vous pouvez obtenir de l'aide sur la commande `passwd` en tapant la commande `man passwd`.
5. Déconnectez vous puis reconnectez vous avec votre nouveau mot de passe.
6. Passez en mode texte puis en mode graphique et revenez en mode texte.
7. Déconnectez vous puis reconnectez vous.

L'environnement standard de travail

Accéder à l'aide

La documentation UNIX est abondante. Elle se présente principalement sous la forme de manuels de référence accessibles par la commande `man`. Mais il existe aussi en lignes (cf. Liens utiles) des HOWTO (« comment faire pour … »), véritables petits modes d'emploi, et des FAQ (« Frequently Asked Questions »), qui regroupent les questions souvent posées et dont les réponses peuvent intéresser beaucoup d'utilisateurs.

La commande man

Cette commande est la principale source d'information sur le système UNIX. Elle informe aussi bien sur les commandes que sur les fichiers de configuration ou sur les primitive de programmation. Les manuels accessibles par cette commande sont organisés en sections. Chaque section est numérotée et traite d'un thème particulier (commandes utilisateurs, commandes d'administration, etc.). Le numéro de section apparaît en tête d'affichage, comme le montre le manuel de la commande `ls` qui permet de lister le contenu d'un répertoire :

```
$ man ls
LS(1)                    Manuel de l'utilisateur Linux                    LS(1)

NOM
       ls, dir, vdir - Afficher le contenu d'un repertoire

SYNOPSIS
       ls [options] [fichier...]
       dir [fichier...]
       vdir [fichier...]

       Options POSIX : [-CFRacdilqrtu1] [--]

       Options GNU (forme courte) : [-1abcdfgiklmnopqrstuvwxABCDFGHLNQRSUX] [-w cols] [-T cols] [-I
       motif]  [--full-time]  [--show-control-chars]   [--block-size=taille]   [--format={long,ver-
```

```
       bose,commas,across,vertical,single-column}]              [--sort={none,time,size,extension}]
       [--time={atime,access,use,ctime,status}] [--color[={none,auto,always}]] [--help] [--version]
       [--]

DESCRIPTION
       La  commande  ls  affiche  tout  d'abord l'ensemble de ses arguments fichiers autres que des
       repertoires. Puis ls  affiche  l'ensemble  des  fichiers  contenus  dans  chaque  repertoire
       indique. Si aucun argument autre qu'une option n'est fourni, l'argument << . >> (repertoire en
       cours) est pris par defaut. Avec l'option -d, les repertoires fournis en  argument  ne  sont
       pas  consideres  comme  des  repertoires  (on  affiche leurs noms et pas leurs contenus). Un
       fichier n'est affiche que si son nom ne commence pas par un point, ou  si  l'option  -a  est
       fournie.
...
```

L'argument fourni à la commande `man` peut pointer sur plusieurs manuels différents. Pour en sélectionner un en particulier, il faut préciser son numéro de section. La commande suivante demande explicitement le manuel de la section 1, pour la commande `mkdir` sous Linux:

```
$ man mkdir
MKDIR(1)
NAME
       mkdir - make directories

SYNOPSIS
       mkdir [OOPTION]... DIRECTORY...

DESCRIPTION
       Create the DIRECTORY(ies), if they do not already exist.

       Mandatory arguments to long options are mandatory for short options too.

       -m, --mode=MODE
              set file mode (as in chmod), not a=rwx - umask

       -p, --parents
              no error if existing, make parent directories as needed

       -v, --verbose
              print a message for each created directory

       -Z, --context=CTX
              set the SELinux security context of each created directory to CTX

       --help display this help and exit

       --version
              output version information and exit
```

Le tableau ci-dessous présente les sections et le type des manuels associés, dans l'ordre de recherche. Quand un manuel est trouvé, la commande `man` l'affiche. La section 8 est intercalée entre la section 1 et la section 2.

Numéro de section	Type de manuel	Exemple
1	Les commandes utilisateurs	`man 1 ls`
8	Les commandes d'administration	`man 8 useradd`
2	Les appels au système	`man 2 signal`
3	Les primitives programmation C	`man 3 printf`
4	Les fichiers spéciaux, les périphériques	`man 4 console`
5	Le format des fichiers	`man 5 passwd`
6	Les jeux	`man 6 fortune`

7	Divers	man 7 DES
9	Le noyau	man 9

Le contrôle d'affichage

L'affichage du manuel est contrôlé par une commande de pagination qui est généralement more ou less. Le contrôle de l'affichage est géré par les caractères suivants:

- **espace** — Cette touche fait passer à la page suivante dans le manuel (la touche « f » à la même attribution).
- **b (backward)** — Pour afficher la page précédente.
- **entrée** — Pour faire défiler ligne à ligne.
- **p** — Pour retourner à la première page.
- **q** — Pour fermer le manuel.
- **/chaine** — Le caractère « / », suivi par une chaîne, recherche ce texte dans les pages du manuel.
- **n (next)** — Pour poursuivre la recherche précédente.

Les options

L'option -k affiche toutes les rubriques contenant, dans leur libellé, un mot spécifique. Elle se sert de la base de données usuellement appelée « whatis » car consultable par le commande whatis, et crée par la commande makewhatis. Voici le résutat de cette commande sur la chaîne passwd:

```
$ man -k mkdir
mkdir (1)            - make directories
mkdir (2)            - create a directory
mkdirat (2)          - create a directory relative to a directory file descriptor
```

L'option -a affiche, les uns après les autres, tous les manuels disponibles pour le mot clés spécifié:

```
$ man -a mkdir
```

Exercice 1.2 (Les manuels)

1. Afficher tous les manuels portant sur la commande mkdir. Puis uniquement celui de la section 1. Que fait la commande mkdir ?
2. Comment faire pour en savoir plus sur la commande man ?

Le répertoire personnel (homedir)

Une fois connecté, l'utilisateur interagit avec l'interpréteur de commande. Dès lors, il peut *voir* les autres utilisateurs, copier des fichiers, lire son courrier ou exécuter d'autres logiciels. Pour cela, il possède un environnement de travail initial, qu'il peut modifier à volonté. Chaque utilisateur possède en particulier un *répertoire personnel* (*home directory* en anglais) où il peut conserver tous ses fichiers : courriers lus, fichiers multimédia, préférences système, … . Le terme couramment utilisé pour ce répertoire est *homedir*.

Vous devez être à même de différencier le répertoire *homedir*, le répertoire *bureau* et un répertoire de travail

La commande pwd (*print working directory*) affiche le nom du répertoire courant. À la connexion, l'utilisateur se retrouve automatiquement dans son répertoire personnel, et cette commande lui permet de se situer, dans l'arborescence du serveur, l'emplacement de son répertoire.

```
$ pwd
/users/ufr/prof/dupont
```

L'administrateur crée et attribue ce répertoire à chacun selon une politique qui lui est propre, mais en respectant généralement les principes d'UNIX, qui le situent dans le répertoire `/home` ou dans une sous-arborescence de ce répertoire. Le nom `/home` est aujourd'hui standard. Toutefois certains systèmes utilisent encore le répertoire `/users/` comme c'est le cas sur les machines de l'UFR de Maths-Info.

Astuce. Pour revenir directement à son répertoire de login, il suffit de taper la commande `cd` (*change directory*) sans aucun argument. Quels que soient les déplacements précédents, on se retrouve chez soit.

Exercice 1.3 (Le homedir)

1. Affichez le répertoire de votre *homedir*.
2. Est-ce que votre *homedir* respecte la convention UNIX ?

L'historique des commandes

L'environnement de travail d'UNIX, conserve les commandes saisies dans un fichier historique. Le système d'historique des commandes permet de rappeler les dernières commandes de manière interactive au niveau du terminal, de les éditer, puis de les valider. Une commande permet d'afficher le contenu de l'historique: il s'agit de la commande `history`:

```
$ history
   1     ls
   2     pwd
   3     exit
```

Vous pouvez remonter dans l'historique en utilisant également des touches flèches haut et bas de votre clavier. Les principales commandes permettant de rechercher une commande passée sont données dans le tableau ci-dessous:

Syntaxe	Action
`!!`	Exécute la dernière commande
`!39`	Exécute la commande n°39. Les numéros sont affichés avec la commande `history`
`!-2`	Exécute l'avant-dernière commande
`!passwd`	Exécute la dernière commande commençant par « `passwd` »
`!?essai`	Exécute la commande contenant le texte « `essai` »

Exercice 1.4 (L'historique)

1. A l'aide de la commande `!` faîtes en sorte de retrouver l'exécution d'une commande `passwd`.

L'auto-complétion des commandes

L'auto-complétion désigne le fait que, dans la ligne de commandes, le nom du fichier ou du répertoire est automatiquement complété par le terminal qui se fonde sur le nom des objets présents dans le répertoire de travail ou dans les répertoires accessibles (dans le PATH [https://fr.wikipedia.org/wiki/Variable_d%27environnement#Sous_Unix.2FLinux_:_.24PATH]). Ainsi, lorsque l'utilisateur tape le début du nom, le shell trouve la suite du nom ou propose plusieurs possibilités. Le système de d'auto-complétion est disponible *via* la touche tabulation. Dans l'exemple suivant, celle-ci est

représentée par la notation <TAB>. La première saisi débute par les lettres pas suivies de la touche <TAB>. Le shell affiche alors les commandes dont le nom débute par pas.

```
$ pas<TAB>
passwd  paste
```

L'auto-complétion s'applique également aux fichiers. La première saisie débute par “ls D” puis la touche <TAB> est pressée (la commande ls permet de lister le contenu d'un répertoire spécifique). Le shell affiche la liste des répertoire qui commence par D:

```
$ ls D<TAB>
Desktop Document
```

L'auto-complétion est très utile pour taper rapidement les commandes et les noms de fichiers sans se tromper.

Les utilisateurs connectés

Une fois connecté, chacun peut voir les autres utilisateurs qui travaillent sur la même machine qu'il s'agisse d'une station de travail sous Linux ou d'un serveur UNIX. Néanmoins, il est moins courant de voir plusieurs utilisateurs connectés sur un micro-ordinateur (Mac, PC, ...) que sur un serveur UNIX.

Afin de prendre conscience que le système est multi-utilisateur, considérons que l'utilisateur “if88888” a ouvert une session graphique et que l'utilisateur if99999 a ouvert de session en mode console via <Ctrl> <Alt> F1 et via <Ctrl> <Alt> F2.

La commande who

La commande who affiche la liste des utilisateurs connectés. Chaque ligne correspond à une connexion. Les informations sont présentées en colonnes. Un nom de login qui apparaît plusieurs fois indique que l'utilisateur en question est connecté à partir de plusieurs postes différents ou bien à partir du même poste, sur lequel il a ouvert plusieurs sessions.

```
if99999     tty1         2013-09-03 08:54
if99999     tty2         2013-09-03 08:55
if88888     pts/0        2013-09-03 08:45
```

La première colonne affiche simplement le nom de login de l'utilisateur. La seconde colonne (*Numéro de ligne de communication*) indique généralement un numéro de connexion par le réseau. Les numérotations habituelles sont préfixées par des mots clés comme *pts*. À chaque connexion par le réseau, une ligne de communication est attribuée à l'utilisateur. Un système UNIX ou Linux exploite ce numéro de ligne (comme *pts/0*) pour savoir sur quel écran envoyer les informations à afficher. Si le préfixe est *tty*, il s'agit d'un terminal connecté sur un port série ou en mode console. L'affichage du numéro *tty1* indique que l'utilisateur est connecté sur le terminal connecté au port série numéro 1 (équivalent d'une connexion via <Ctrl> + F1). La troisième colonne (*Date*) indique l'heure de connexion. Elle permet, par exemple, de constater qu'un utilisateur a laissé sa session ouverte depuis... la veille ! ou encore l'heure à laquelle il a débuté sa séance de TP. L'administrateur peut éventuellement le déconnecter à distance en « tuant » sa session avec la commande kill que nous verrons dans les séances suivantes. Finalement, la colonne n°4 (*Adresse du poste de travail*) « localise » l'utilisateur. Lorsque le nom est indiqué, il est possible de savoir où il se trouve. A ce stade les utilisateurs sont locaux, nous verrons dans les TP suivants que les utilisateurs peuvent être distants.

Astuce. La commande who fournit beaucoup d'informations sur les utilisateurs connectés. L'administrateur y a souvent recours pour récupérer un utilisateur (adresse de poste de travail), vérifier depuis quand il est

connecté (date et heure de connexion) ou afficher un message sur son écran. Dans ce cas, il peut se servir des commandes `write` ou `wall`, qui envoient un message sur l'écran de l'utilisateur (`write`) ou de tous les utilisateurs (`wall` pour *write all*).

Astuce. Pour ne plus recevoir de message taper la commande:

```
$ mesg n
```

La commande `who` possède également une autre syntaxe qui permet d'afficher uniquement la ligne qui concerne l'utilisateur, `who am i`. Voici un exemple de son exécution:

```
if88888  pts/0        2010-08-12 11:10 (mars.math-info.univ-paris5.fr)
```

Astuce. Pour toutes les commandes système, il existe une aide en ligne qui peut être appelée au moyen de la commande `man` (pour */manual*), suivi du nom de la commande pour laquelle on souhaite de l'aide.

La commande w

Les informations fournies pas le commande `w` sont semblable à celles de la commande `who`. Le changement principale est lié à la dernière colonne, qui affiche la commande que l'utilisateur exécute à la place de l'adresse du poste de travail. Ainsi chaque personne connectée peut voir ce que font les autres. Dans l'exemple ci-dessous, l'exécute la commande `w`.

```
$ w
 11:16:38 up 26 days, 23:40,  1 user,  load average: 0,00, 0,00, 0,00
USER     TTY      FROM              LOGIN@   IDLE   JCPU   PCPU WHAT
if88888  pts/0    mars.math-info.u 11:10    0.00s  0.04s  0.00s w
```

L'entête présente l'heure actuelle (11:16:38), indique depuis quand le système est opérationnel (up 26 days), le nombre d'utilisateurs connecté (1 user) et la charge moyenne du processeur pour les 1, 5, 15 dernières minutes.

La commande finger sans argument

Cette commande possède deux syntaxes qui se ressemblent mais qui donnent des résultats différents. Dans la syntaxe présentée ci-dessous, la commande `finger` sans argument, affiche les informations de type « état civil » pour chaque utilisateur connecté, sinon elle affiche seulement les informations concernant l'utilisateur en paramètre. À côté du nom de login, dans la colonne « Name » apparaissent le prénom et le nom de l'utilisateur et dans la colonne « Site Info » les autres informations que l'administrateur a enregistré lors de la création du compte de l'utilisateur.

```
Login     Name      Tty           Idle  Login Time   Office     Office Phone
if88888   Dupont    pts/0         Aug 12 11:10 (mars.math-info.univ-paris5.fr)
```

La seconde syntaxe de la commande utilise un argument. Elle informe sur l'identité de la personne dont le nom de login est fourni, même si cette personne n'est pas connectée. Cette seconde syntaxe est étudiée en détail à la section les commandes d'identification

Le filtrage de l'affichage

Les commandes `who`, `w` et `finger` fournissent des informations intéressantes sur les utilisateurs connectés. Sans bénéficier d'aucun privilège particulier, chacun peut savoir de quel poste et depuis

combien de temps quelqu'un travail (who), ce qu'il fait (w) et comment il s'appelle (finger). Ces commandes de bases facilitent le travail de surveillance des administrateurs et des enseignants. Elles leurs permettent d'analyser simplement et rapidement l'activité du serveur en regardant qui fait quoi.

Ainsi, la dégradation du temps de réponse d'un serveur peut être due à un trop grand nombre d'utilisateurs connectés, ou à l'utilisation d'un logiciel qui charge fortement le processeur. Un moyen simple de le vérifier consiste à comptabiliser le nombre d'utilisateurs actifs ou le nombre d'exécutions du logiciel en question. Quand l'information fournies est bien ciblée, il est facile d'intervenir, soit en changement la priorité des processus (logiciels en cours d'exécution), soit en déconnectant les gros consommateurs (dans le cas extrême où un ralentissement bloque le système). Il est aussi possible d'aller voir ces utilisateurs si l'adresse de leur poste de travail montre qu'ils sont géographiquement proches ou encore leur envoyer un message avec la commande write.

Nous allons nous intéresser ici à enchaîner des commande de filtrage et de comptabilisation aux commandes précédentes pour un obtenir un résultat plus précis et plus facilement exploitable. Les deux commandes utilisées sont grep et wc. L'enchaînement de commandes utilise le caractère « | » (prononcer « païpe », pour « pipe » en anglais). l'enchaînent des commandes who avec la commande wc se note:

```
$ who | wc
```

Dans cet exemple que vous pouvez essayer, l'affichage de la commande who n'apparaît plus à l'écran; il est redirigé vers la commande wc. Cette commande traite le texte qu'elle reçoit et affiche le résultat à l'écran.

La commande grep. La commande grep filtre le flux de texte qu'elle reçoit et ne laisse passer que les lignes contenant la chaîne de caractères donnée en argument. Dans l'exemple ci-dessous, seules lignes de texte fournies par la commande who et contenant le text « dupont » s'affiche à l'écran:

```
$ who | grep math-info.univ-paris5.fr
dupont  pts/0        2010-08-12 12:15 (mars.math-info.univ-paris5.fr)
```

Cette ligne de commandes donne rapidement la liste des utilisateurs connecté sur la machine depuis le domaine « math-info.univ-paris5.fr ». Toute les lignes contenant le texte utilisé comme critère de filtrage s'affichent, quel que soit l'emplacement du texte sur la ligne (début, milieu ou fin).

Le signe « ^ » en début de critère indique que le texte concerné doit se trouver en début de ligne. Il faut alors utiliser les guillemets. Ainsi, la syntaxe suivante affiche toutes les lignes commençant par « dupont »:

```
$ who | grep "^dupont"
dupont  pts/0        2010-08-12 12:15 (mars.math-info.univ-paris5.fr)
dupont  pts/1        2010-08-12 12:32 (saphyr.ens.math-info.univ-paris5.fr)
```

Cette ligne de commandes affiche toutes les connexions de l'utilisateur « dupont ».

Astuce. Vous pouvez utiliser le caractère ($) qui, lui, indique la fin de la ligne: who | grep “fr)$” renvoie toutes les lignes terminant par « fr) ».

Si l'utilisateur veut savoir qui utilise le logiciel firefox, il suffit de taper la ligne de commande:

```
$ w | grep firefox
dupont   pts/5     04:13PM          44         28:19          38:19          ./firefox
martin  pts/19    04:36PM           0          1:09           1:09          ./firefox
```

Ici c'est le résultat de la commande w qui est exploité pour afficher la liste des utilisateurs qui utilisent un logiciel particulier.

La commande wc. Cette commande, à la consonance particulière en français, compte le nombre de caractères, de mots ou de lignes d'un texte (*wc* est l'abréviation de *word coun*). Les options `-l`, `-w` et `-c` précisent de n'afficher respectivement que le nombre de lignes, de mots et de caractères. L'affichage du nombre d'utilisateurs connectés s'obtient simplement en tapant:

```
$ who | wc -l
```

Voici deux exemples où les commandes `grep` et `wc` sont utilisées simultanément. La syntaxe suivant affiche le nombre d'utilisateurs exécutant le logiciel « firefox ».

```
$ w | grep firefox | wc -l
          2
```

Dans cette exemple, le résultat de la commande `w` est transmis à la commande `grep`, qui filtre les lignes contenant le text « firefox ». Le résultat de la commande `grep` est comptabilisé par la commande `wc -l`. Il y a deux utilisateur utilisant le logiciel « firefox ».

La syntaxe suivante affiche le nombre d'utilisateurs qui travaillent à partir de poste se trouvant dans la salle `523a1`:

```
$ who | grep 523a1 | wc -l
         15
```

L'enchaînement de commandes n'est pas limité pat leur nombre mais pas la signification des données à traiter pour la nouvelle commande. Dans l'exemple précédent, la dernière commande affiche la valeur « 15 ». Enchaîner avec une nouvelle commande `grep` n'aurait aucun sens !

Exercice 1.5 (Les utilisateurs connectés en local)

1. Afficher et lire l'aide la commande `who`. Donner une description sommaire de ce que fait la commande `who`.
2. Taper la commande `whoami` et détailler les informations obtenues.
3. Taper la commande `who`. Combien il y a d'utilisateurs connecté ? Vous pourrez utiliser la commande `wc`.

Le réseau

Secure Shell (ssh)

Le Secure shell (ssh) est utilisé pour ouvrir un shell sécurisé (console) sur un ordinateur distant. Il est basé sur le protocole éponyme qui permet une communication encryptée entre un serveur et un client. L'encryptage améliore la sécurité de cette communication car il rend difficile l'interception et l'analyse des informations qui transitent par le réseau.

Habituellement, un serveur ssh attend les requêtes clients sur le port 22.

Connexion à distance

Pour se connecter à une machine distante, il faut utiliser la commande

```
$ ssh serveur
```

ou :

```
$ ssh utilisateur@serveur
```

Si le port du serveur n'est pas le port par défaut, on utilisera l'option `-p` :

```
$ ssh -p port utilisateur@serveur
```

Ensuite, le serveur procédera à une authentification en vous demandant votre mot de passe. Si vous vous connectez pour la première fois à cette machine, cela devrait donner :

```
[moi@local]$ ssh utilisateur@serveur
The authenticity of host 'serveur (127.0.0.1)' can't be established.
RSA key fingerprint is 44:77:2a:22:d7:07:80:a9:e3:a6:df:33:8b:b3:78:0a.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'serveur' (RSA) to the list of known hosts.

utilisateur@serveur's password :
Last login: Mon Jan  1 12:00:21 2010 from 192.168.0.1
[utilisateur@serveur]$
```

Lorsque vous vous reconnecterez à cette machine vous aurez un rappel de votre dernière connexion, ainsi que le nom ou l'adresse IP de la machine que vous aviez utilisé pour vous connecter.

```
[moi@local]$ ssh utilisateur@serveur
utilisateur@serveur's password :
Last login: Mon Jan  1 12:00:21 2010 from 192.168.0.1
[utilisateur@serveur]$
```

Remarque : Lors d'une connexion en ssh, à la place du nom du serveur, on peut aussi utiliser son adresse IP. C'est parfois utile lorsque le service de résolution de nom (DNS) pose problème.

Lorsque vous vous connectez depuis l'extérieur de l'UFR, prenez garde à bien indiquer le nom **complet** de la machine sur laquelle vous souhaitez vous connecter.
Exemple pour saphyr : **saphyr.ens.math-info.univ-paris5.fr**

Pour clore la session on utilise la commande suivante :

```
$ logout
```

ou:

```
$ exit
```

On peut également utiliser le clavier avec la combinaison :

```
 <Ctrl>+D
```

Cette combinaison n'est ni un raccourci clavier (qui dépend d'un environnement ou d'un programme) , ni un signal (contrairement à `<Ctrl>+Z` par exemple). Il s'agit en fait de produire un caractère spécial de la table ASCII appelé *fin de fichier*. Ce caractère est désigné par un acronyme : `EOT` (End Of Text). En utilisant ce caractère sur l'entrée standard du terminal, on met fin au shell.

Vous trouverez toute la table ASCII ici

Copie de fichiers à distance

De la même manière, la commande `scp` (pour Secure Copy) permet de copier des fichiers et des arborescences entre deux machines, en utilisant le protocole ssh afin de sécuriser les transferts de fichiers.

Pour réaliser la copie d'un fichier sur une machine distante, il faut utiliser la commande:

```
$ scp ./mon_fichier utilisateur@serveur:chemin/
```

Pour réaliser la copie d'une arborescence (récursivement) depuis une machine distante, il faut utiliser la commande:

```
$ scp -r utilisateur@serveur:chemin/ ./mon_repertoire/
```

Ensuite, le serveur procédera à une authentification en vous demandant votre mot de passe.

L'option `-p` permet de conserver les dates de modification, d'accès et les permissions des fichiers originaux.

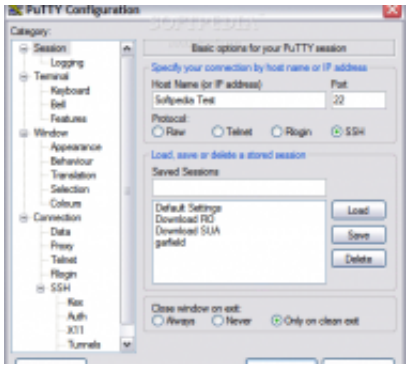
```
$ scp -r utilisateur@serveur:chemin/ ./mon_repertoire/
```

Cette commande peut vous servir lorsque vous travaillez en groupe; à la fin d'un TP chaque membre du groupe peut recopier les fichiers créés au cours de la séance dans son répertoire personnel (home directory).

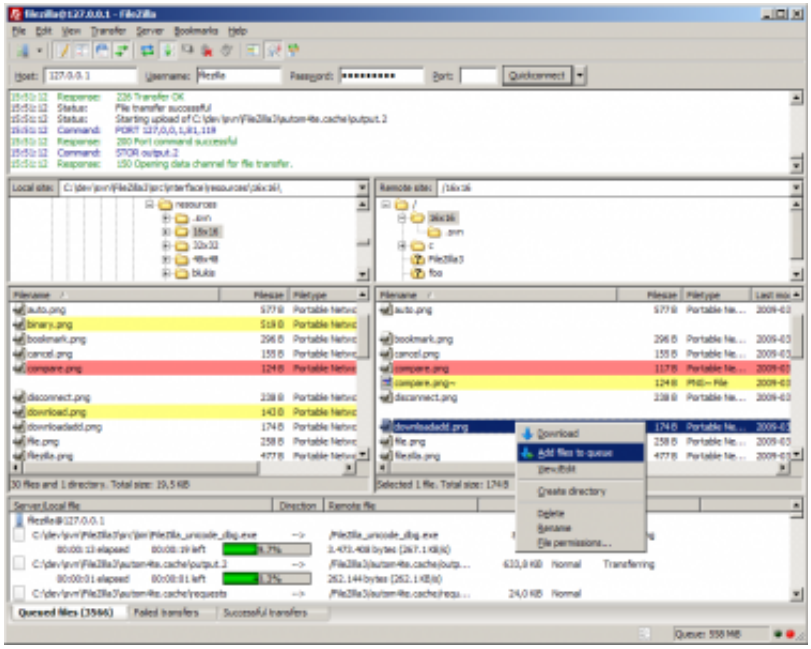
Sous Windows ...

Vous pouvez vous connecter à distance sur une machine unix/linux pourvue d'un serveur ssh si vous possédez un compte (login) sur cette machine. Et ce, même depuis une machine qui utilise Windows comme système d'exploitation. Pour cela il vous faut un client ssh.

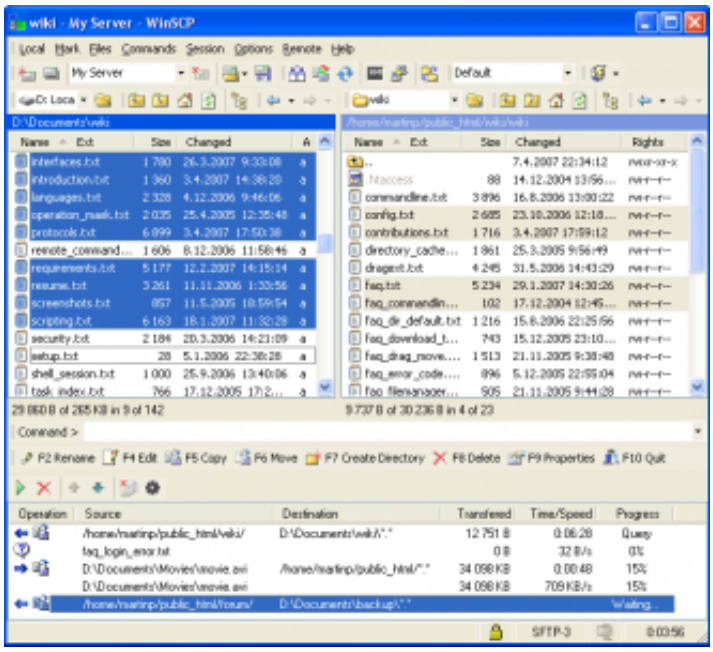
Pour lancer un terminal, nous vous conseillons d'utiliser PuTTy [http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html]. Il s'agit d'un client ssh open-source [http://fr.wikipedia.org/wiki/Open_source] (et donc gratuit) qui propose de nombreuses options de configuration.



Pour récupérer vos fichiers (si vous avez oublié votre clé USB par exemple), nous vous conseillons d'utiliser FileZilla [http://filezilla-project.org/download.php?type=client]. Il s'agit d'un client ftp, compatible sftp (un protocole de transfert de fichiers sécurisé basé sur le protocole ssh). Ce logiciel est aussi en open-source.



Une autre solution, consiste à utiliser le logiciel open-source [http://fr.wikipedia.org/wiki/Open_source] WinSCP [http://winscp.net/eng/download.php].
Il compatible avec les protocoles scp, sftp et ftp.



Pour pouvoir utiliser ces outils dans de bonnes conditions, nous vous recommandons de bien lire la documentation de ces programmes. En outre, n'hésitez pas à chercher dans la FAQ si votre question n'a pas déjà sa réponse. Nous ne pourrons pas configurer vos connexions.

L'ancêtre de MSN/Skype

La commande `talk` permet de discuter avec un autre utilisateur connecté sur la même machine que vous. Vous pouvez utiliser cette commande si le démon `talkd` est lancé et que sa configuration vous y autorise.

talk n'est plus installé sur les machines de l'UFR 😕

Exercice 1.6 (Les utilisateurs connectés à distance)

Vous aller pour le reste de l'exercice vous connecter à une machine distante. Cette machine à pour adresse `saphyr`. Cette machine est le serveur qui stocke vos fichiers. Pour vous connecter, tapez la commande `ssh -X saphyr` puis saisissez votre mot de passe. Vous êtes maintenant connectés sur `saphyr`.

1. Lancer à nouveau la commande `who`. Combien d'utilisateurs sont maintenant connectés. Vous pourrez utiliser la commande `wc`.
2. Lancer la commande `who` en affichant que les lignes vous concernant. Vous pourrez utiliser la commande `grep`. Depuis quelle machine êtes vous connectée ?
3. Repérer le login et le terminal de votre voisin à l'aide la commande `who`, puis envoyer lui des messages grâce à la commande `write`. Pour vous aider sur la syntaxe de la commande `write`, nous vous rappelons que vous pouvez obtenir de l'aide en tapant `man write`.
4. Taper la commande `w` et donnez les informations suivantes:
   I. depuis quand le système est opérationnel
   II. le nombre d'utilisateur connecté en ce moment
   III. la charge moyenne du processeur les 1, 5 et 15 dernières minutes.
   IV. si votre voisin est connecté que fait-il ?
5. Utiliser la commande `finger` pour obtenir les informations nom et prénom des utilisateurs connectés. Quel est le login de votre voisin ? Si vous connaissez son nom ou son prénom vous pouvez filtrer en utilisant la commande `grep`

L'identification d'un utilisateur

Sur un système UNIX, l'identification repose sur un login authentifié par un mot de passe. C'est grâce à ce nom que le système reconnaît l'utilisateur et ouvre la session de travail. Cette identification de l'utilisateur à un nom de login correspond à une vision simplifiée du fonctionnement du système. En fait, chaque utilisateur connecté est reconnu grâce à un numéro interne au système UNIX, et non pas son nom de login.

Principe

Le système UNIX est multi-utilisateur, c'est-à-dire que plusieurs personnes peuvent se connecter simultanément. Cela implique que chaque utilisateur soit clairement identifié, et que les permissions qui lui sont associées soient définies. Pour cela, UNIX s'appuie sur deux notions, l'*identifiant* et le *groupe d'appartenance*.

La notion d'identifiant

Un système UNIX utilise un numéro pour reconnaître un utilisateur. Ce numéro, l'*UID* (*User IDentification*), est le seul identifiant reconnu par le système. Le nom de login est là uniquement pour simplifier le travail de l'utilisateur, comme le nom d'ordinateur simplifie la syntaxe des adresses réseaux en se substituant au adresse IP.

À l'ouverture de session, l'utilisateur fournit un nom de login également appelé *UNAME* (*User NAME*). Le système cherche ce nom dans un fichier contenant les informations de tous les utilisateurs enregistrés, pour y trouver l'UID correspondant. Ce fichier se nomme `/etc/passwd` (prononcer `etc` puis `passwd`). Pendant toute la session de travail, l'utilisateur est connu grâce à son UID. Les droits de lecture d'écriture sur les fichiers ou répertoires, les droits d'exécution de tel ou tel logiciel dépendent uniquement de son identité, c'est-à-dire son UID. L'administrateur attribue un UID différent à chaque utilisateur lors de son enregistrement. Si un même numéro est affecté à deux utilisateurs distinct, ils sont assimilés à un seul utilisateur lorsqu'ils sont connectés.

La notion de groupe d'utilisateurs

UNIX attribue également un numéro de groupe d'utilisateur, le *GID* (*Group Identification*) à chacun. Un utilisateur peut gérer les droits sur ses fichiers en autorisant, par exemple, la lecture à ceux qu'il « connaît » (les membres de son groupe), et en ne donnant aucun droit aux autres. La notion de groupe autorise cette gestion des droits. Chaque utilisateur appartient à un *groupe principal* et peut être membre d'autre groupes.

Les commandes d'identification

Les informations système sont obtenues par la commande `id`, tandis que la commande `finger` informe sur l'état civil de l'utilisateur.

La commande id. Deux syntaxes sont généralement utilisées pour la commande `id`. La première retourne les informations sur l'utilisateur qui tape la commande:

```
uid=4910(dupont) gid=401(ufr)
```

Le nom de login (ou UNAME) de cet utilisateur est `dupont`, et son UID est `4919`. Il appartient au groupe `ufr` qui porte le numéro `401`. La commande `id` trouve ses informations dans les fichiers `/etc/passwd` et `/etc/group`.

La seconde syntaxe retourne les informations sur un utilisateur dont le nom de login est fourni en argument:

```
id martin
uid=4751(martin) gid=401(ufr) groupes=10001(svnL2201001),10002(svnL2201002)
```

Tous les utilisateurs peuvent obtenir cette information. Elle indique que `martin` possède l'UID `4751` et que son groupe principal et le `401`, dont le nom est `ufr`. De plus `martin` est membre de deux autres groupes `svnL2201001` et `svnL2201002` dont les numéros sont respectivement `10001` et `10002`.

Chaque fichier possède un *propriétaire* et un *groupe propriétaire*. Le propriétaire est le créateur et le groupe propriétaire le groupe du créateur au moment de sa création. Les fichiers de notre utilisateur `martin` auront comme propriétaire `martin` et comme groupe `ufr`.

La commande finger avec argument. La commande `finger` tapée sans argument affiche des informations sur les utilisateurs connectés.

Quand elle est utilisée avec un nom de login comme argument, `finger` affiche des informations sur l'utilisateur spécifié, tel qu'il est enregistré sur le serveur. Cette affichage n'est pas liés à la connexion de l'utilisateur, mais à son enregistrement par l'administrateur. Comme la commande `id`, la commande `finger` trouve les informations concernant le login indiqué dans le fichier `/etc/passwd`. Elle affiche le nom, le prénom et d'autre informations concernant l'utilisateur. Voici un exemple d'utilisation:

```
$ finger dupont
Login: dupont                           Name: Dupont
Directory: /users/ufr/prof/dupont       Shell: /bin/bash
On since jeu. aout 12 12:15 (CEST) on pts/0 from mars.math-info.univ-paris5.fr
Last login jeu. aout 12 12:32 (CEST) on pts/1 from saphyr.ens.math-info.univ-paris5.fr
No mail.
No Plan.
```

Exercice 1.7 (Identification)

1. Répondez aux questions de révisions suivantes:
   - Comment est identifié un utilisateur ?
   - Quel est le fichier qui permet d'identifier un utilisateur ?
   - Quel est le rôle du fichier `/etc/group`
2. Affichez les informations (uid, gid) vous concernant.
3. Affichez les informations (uid, gid) concernant votre voisin.

Les alias

Les *alias* font partie des fonctionnalités intéressantes d'Unix. Leur mise en oeuvre permet un gain de temps immédiat. En effet, durant une session de travail, un certain nombre de commandes reviennent souvent. Quand leur syntaxe est simple, cela n'a pas d'importance, mais quand elle est complexe ou longue, un raccourci est apprécié. C'est justement le but des alias, qui permettent à chaque utilisateur de créer ses propres commandes à partir de commandes existantes. Ces “raccourcis” ont une durée de vie limitée à celle de la session. Pour les conserver, il faut les recréer à chaque nouvelle session, et c'est le rôle des fichiers de démarrage.

La commande alias sans argument affiche la liste de tous les alias définis.

Dans le répertoire /etc/profile.d est situé un fichier aliases.sh que les administrateurs réseaux ont crées et dont voici le contenu :

```
alias mv='mv -i'
alias rm='rm -i'
alias cp='cp -i'
alias c=clear
alias j='jobs -l'
alias lt='ls -alt -r --color'
alias h=' history|more'
alias Rm='/bin/rm'
alias smbpasswd='/usr/bin/smbpasswd -r diamant'
alias mysql='mysql -h opale'
```

On redéfinit par exemple les commandes cp, mv et rm en leur donnant le paramètre « -i » qui signifie interactif : l’utilisateur doit ainsi valider la suppression d’un fichier s’il utilise la version « aliassée » de la commande `rm`. En revanche, s’il tape :

```
$ \rm fichier
```

alors c’est la « vraie » commande rm qui est exécutée, le système ne prend plus en compte l'alias et dans ce cas la suppression du fichier est irrémédiable.

Tout utilisateur peut créer un alias en l’ajoutant dans son fichier .bashrc.

exemple :

```
alias bjour='echo Bonjour nous sommes le $(date)'
```

Cet alias, disponible en tapant simplement :

```
$ bjour
```

dans un terminal. Cet alias affiche le texte « Bonjour nous sommes le : : » suivi de la valeur retournée par la commande `date`.

Notez que l’on exécute en fait deux commandes séparées par le signe ” ;”.

Lorsque l'on retape une commande alias avec le même nom pour l'alias, celui-ci est alors modifié.

La commande de supression de l'alias est unalias, suivi du nom de l'alias à supprimer. La commande suivante retire l'alias bjour :

```
unalias bjour
```

Exercice 1.8 (Les alias)

1. Créer un alias pour la commande `history` qui se nommera `h`, puis vérifier que votre alias fonctionne.
2. Tapez la commande `alias` dans l'invite de commande pour voir tous les alias déjà définis pour votre session. Regardez le manuel pour les commandes inconnues de la deuxième partie de l'alias
3. Ecrivez la commande `psmoi` permettant d'obtenir la liste de tous les processus vous appartenant (utilisez la commande `ps`)
4. Testez la commande : `alias bjour="echo Bonjour nous sommes le $(date)"`. Que fait elle ?
5. Quelle est la commande à taper pour vérifier que ces alias ont bien été pris en compte?
6. Editer le fichier `.bashrc` avec `gedit` par exemple

```
$ gedit .bashrc &
```

puis ajouter votre alias à la fin du fichier puis taper la commande

```
$ source .bashrc
```

pour demander au terminal de lire à nouveau le fichier de configuration. Fermer et relancer votre terminal. Vérifier que votre alias est toujours valide.

La gestion des processus

Tout logiciel est à la base un programme constitué d'un ensemble de lignes de commandes écrites dans un langage particulier appelé langage de programmation. C'est uniquement quand on exécute le logiciel que le programme va réaliser la tâche pour laquelle il a été écrit, dans ce cas là on dira qu'on a affaire à un processus ou process. En d'autres termes le programme est résolument statique, c'est des lignes de code, alors que le process est dynamique, c'est le programme qui s'exécute.

Par exemple le logiciel Word sous Windows est en fait un bête programme écrit dans un langage qui a été ensuite compilé pour le rendre compréhensible par la machine, ce n'est uniquement que quand vous le lancez, que vous avez alors affaire au process Word.

Ainsi, on peut avoir à un moment donné plusieurs processus en cours, à un temps donné. Le système doit être capable de les identifier. Pour cela il attribue à chacun d'entre eux, un numéro appelé PID (Process Identification).

Un processus peut lui même créer un autre processus, il devient donc un processus parent ou père, et le nouveau processus, un processus enfant. Ce dernier est identifié par son PID, et le processus père par son numéro de processus appelé PPID (Parent Process Identification).

Tous les processus sont ainsi identifiés par leur PID, mais aussi par le PPID du processus qui la créé, car tous les processus ont été créés par un autre processus. Oui mais dans tout ça, c'est qui a créé le premier processus ? Le seul qui ne suit pas cette règle est le premier processus lancé sur le système le processus init qui n'a pas de père et qui a pour PID 1. Son rôle est de lancer l'exécution des sous-systèmes, comme

les systèmes de fichiers, le réseau, etc. Chaque sous-système va à son tour activer des logiciels qui vont prendre en charge des focntionnalités précises du système UNIX.

Affichage des processus : les commandes ps et top

La commande ps affiche les informations sur les processus actifs. Par défaut, elle présente la liste des processus attachés à la connexion. Lorsque cette commande est saisie sans argument, elle comporte les colonnes suivantes:

- PID : numéro du processus
- TTY : numéro de ligne de communication associée au processus
- TIME : temps total d'éxécution de la commande
- CMD : commande associée au processus

À la commande ps sont associées des options qui permettent d'afficher davantage d'informations. Les principales sont les suivantes :

- -l Affiche des informations plus complètes sur les processus. Elle présente des colonnes supplémentaires telles que :
  - UID : numéro UID de l'utilisateur
  - PPID : numéro du processus père
  - PRI : niveau de priorité du processus. Plus le nombre est grand, plus la priorité est faible.
  - NI : valeur de “gentillesse” traitée par la commande nice, qui sert à calculer la priorité. Une valeur positive indique un accès moindre aux ressources du processeur.
  - S : état du processus. La lettre S indique que le processus “dort” (Sleeping), R ou A précise qu'il est en éxécution (Running ou Active).
- -f Affiche des informations un peu moins complètes qur l'option -l mais plus lisibles. La nouvelles colonne STIME indique l'heure de début du processus.
- -a Affiche les processus des autres utilisateurs
- -e Affiche la liste de tous les processus sauf ceux du noyau
- -u Affiche les informations sur l'utilisateur dont le nom est spécifié. Une liste de nom peut être fournie telle que

```
ps -u martin,dupont
```

- -t Affiche les informations sur les processus associés à un terminal. Une liste de noms peut être fournie
- -o Permet de sélectionner le format d'affichage dans l'ordre voulu selon une liste de mots clés prédéfinis, qui peuvent varier d'un système à l'autre, mais dont les principaux sont :
  - user : nom de l'utilisateur
  - group : nom du groupe du processus (généralement le groupe de l'utilisateur)
  - uid : numéro de l'utilisateur
  - gid : numéro du groupe du process - Liste numérotéeus
  - pid : numéro du processus
  - ppid : numéro du processus père
  - command : nom de la commande
  - tty : nom de la ligne du terminal
  - nice : valeur gérée par la commande nice
  - time : temps cumulé du processus

Ces mots-clés sont à séparer par une virgule

La commande `top` permet de visualiser de manière dynamique les processus présents sur la machine ainsi que le taux de charge du processeur et le taux d'occupation de la mémoire.

Gestion des processus interactifs, avant-plan et arrière-plan

Quand un utilisateur exécute une commande interactive, il doit attendre qu'elle soit terminée pour en exécuter une autre au niveau d'une même fenêtre de terminal. C'est le fonctionnement normal du shell, qui exécute par défaut les tâches en "avant-plan" (foreground). Il est possible d'éviter le blocage de la ligne de commande en lançant directement la tâche en "arrière-plan" (background). Dans ce cas, le processus s'exécute sans bloquer l'invite de commande, et l'utilisateur peut saisr une nouvelle commande sans attendre la fin de la première. Le nombre de processus en arrière-plan n'est pas limité, alors qu'il ne peut y avoir qu'un seul processus en avant-plan. Le lancement d'une ligne de commande en arrière-plan utilise le signe *&* à la fin de la ligne. Voici un exemple de deux syntaxes qui accomplissent la même action. La première est exécutée en avant-plan, la seconde utilise le signe *&*, elle est donc lancée en arrière-plan.

```
$ sleep 60
$ sleep 60 &
```

Il est toujours possible de faire basculer une tâche d'avant en arrière, et vice-versa, avec les commandes `fg` (pour foreground) et `bg` (pour background). Pour passer en arrière-plan une tâche lancée en avant-plan, il faut tout d'abord la suspendre sans l'arrêter avec la combinaison CTRL+Z.

```
$ sleep 60
^Z
[1]+ Stopped     sleep 60
```

La commande `bg` reprend l'exécution en arrière-plan:

```
$ bg
[1]+    sleep 60 &
```

La commande jobs confirme que la commande continue son exécution

```
$ jobs
[1]+    Running    sleep 60 &
```

La commande `fg` passe la commande en avant-plan, ce qui rend à nouveau la saisie impossible. La commande est arrêtée avec CTRL+C.

```
$ fg
$ sleep 60
^C
```

Attention. La suspension d'un programme s'effectue par la combinaison de touches CTRL+Z. L'arrêt d'un programme s'effectue par la combinaison de touches CTRL+C.

La commande kill

La commande `kill` envoie un signal aux processus actifs. Le numéro du signal et le PID du processus sont indiqués sur la ligne de commande. Si aucun numéro de signal n'est précisé, c'est SIGTERM (15) qui est envoyé. Ce signal tue les processus, sauf ceux qui l'interceptent. Pour ces processus, il faut envoyer explicitement le signal SIGKILL (9), qui ne peut être intercepté. L'administrateur *root* peut arrêter tous les processus avec la commande kill. Les autres utilisateurs ne peuvent gérer que les processus dont ils sont propriétaires.

Les différentes syntaxes de cette commande sont :

```
$ kill -num_signal pid1 pid2 etc.
$ kill -s num_signal pid1 pid2 etc.
$ kill -l
```

où num_signal est le numéro ou le nom de signal à envoyer, et `pid1`, `pid2`, etc., sont les numéros de processus vers lesqueks envoyer le signal. Les PID des processus sont obtenus avec la commande ps. La commande kill avec l'option `-l` affiche la liste et le nom des signaux qui peuvent être administrés.

Exercice 1.9 (Les processus)

1. Affichez la liste des signaux connus
2. Qu'est ce que le PID et le PPID d'un processus?
3. Quelle est la différence entre la commande `ps -f` et la commande `ps -ef`?
4. Quelle commande faut-il saisir pour obtenir la liste des processus de l'utilisateur martin? Connectez vous à `saphyr` comme dans l'exercice 1.1, choisissez un utilisateur connecté et lister les processus de cet utilisateur ?
5. Affichez la liste de vos processus avec les informations suivantes : l'UID, le PID, le PPID et la commande, dans cet ordre
6. Lancez la commande `sleep 120` en arrière plan. Repérez le processus grâce à la commande `ps` et tuez ce processus.
7. Lancez la commande man mkdir dans un terminal. Suspendez cette commande et repérez cette commande dans votre liste de processus. Tuez cette commande.
8. Testez la suite de commandes décrite dans la partie Gestion des processus interactifs, avant-plan et arrière-plan.