



LINUX

Système, administration et services réseaux

Introduction au système

- [L'environnement graphique KDE](#)
- [Exemple d'installation](#)
- [Démarrer \(sous\) Linux](#)
- Organisation du [système de fichiers](#)
- Principales [commandes utilisateurs](#)
- [TP Extension du système](#)
- [Installation d'applications et archivage](#)
- [Le service d'impression](#)
- Installation et réglages du [serveur X](#)
- [Outils d'administration](#) [TP Webmin](#)
- [Suivi et planification des processus](#)
- [Configuration du réseau](#)
- NFS : [présentation](#) & [installation d'une station](#)
- [Références et documentation Internet](#)

Gestion des utilisateurs & protection des fichiers

Outils TCP/IP et services réseaux

- TCP/IP, [commandes et outils de base](#)
- [TCP Wrappers](#), contrôler l'accès aux services
- [Syslog](#), pour gérer les journaux
- [Crontab - at](#), pour planifier les tâches
- [Ipchains](#), comprendre le principe d'un firewall
- [Service DHCP](#), attribution automatique d'adresses IP
- [Correction des exercices](#)
- [Bind](#), serveur DNS
- [Squid](#), proxy-cache
- Présentation du [projet SLIS](#), "Serveur Linux pour l'Internet Scolaire"

Courrier à [Philippe Chadeaux](#)

- [Gestion des utilisateurs](#)
- [Permissions des fichiers](#)
- [TP de synthèse](#)
- [TP corrigé](#)

Stations Windows en réseau Linux

- [Installation et utilisation de SaMBa](#)
- [TP prise en main de Samba](#)
- [Exemple de session de travail](#)
- [Compléments](#)
- [Samba, contrôleur de domaine](#)
- [Mise en place d'une messagerie locale](#)
- [Présentation de SambaEdu](#)

Téléchargement

-  [format TGZ](#) (installation : tar xzvf cours-linux.tgz)
-  [format ZIP](#)

Courrier à [Jean Gourdin](#)

Shell et programmation

Shell et scripts BASH

- [L'interpréteur de commandes](#)
- [Introduction aux scripts](#)
- [TP1 exercices](#) -- [corrigé](#)
- [TP2 créer des comptes](#) -- [corrigé](#)
- [Initiation aux expressions rationnelles](#)
- [Approche des filtres](#)

Langage PERL

- [Introduction à Perl](#)
- [Les expressions rationnelles en Perl](#)
- [Traitement d'un formulaire](#)

Autres supports de cours

- [Introduction à Java et à la programmation-objet](#)
- [Introduction à JavaScript](#)

Courrier à [Jean Gourdin](#)

Réorganisation et mise à jour le 5 janvier 2001



Découverte de X-KDE

*KDE = K Desktop Environment, environnement de travail graphique sous X
projet récent (96), devenu rapidement stable, et en plein développement (bientôt KOffice).*

Généralités

Examen de la documentation

- Activer le lanceur **doc** du bureau --> page d'accueil `/usr/doc/mandrake/index.html`
- choisir en fr, *Mandrake/ userguide* pour consulter le guide l'utilisateur de la distribution
- choisir en fr, *KDE/ userguide* introduit à la prise en main de KDE
- lire notamment 2. *Introduction*

Important

- Si une application graphique se "plante" on garde normalement la main sur les autres !
--> pour forcer sa fermeture, on la "tue". Pour cela 2 procédés :
 - on clique sur l'icone du bureau **Xkill**, puis on sélectionne la fenêtre de l'application récalcitrante.
 - si l'icone précédente est inaccessible, on ouvre un terminal **Konsole** et on lance l'utilitaire **xkill**.
- Le système graphique ne répond plus !
Le problème vient le plus souvent d'une mauvaise [configuration du serveur X](#)
--> Ctrl-Alt Fx pour revenir au terminal sttyx dans lequel on a lancé X, puis Ctrl-C
--> on peut aussi activer un autre terminal en mode texte, ou en ouvrir Ctrl-Alt-F(x+1)
--> on peut revenir au serveur X, par Ctrl-Alt F7
Enfin débarrassé du funeste écran bleu, avec son message cynique ("l'appli xxx a provoqué une exception dans le ... ").

Principales fonctionnalités de KDE

- KDE présente à l'utilisateur 4 bureaux *virtuels*, par défaut, avec lesquels il peut travailler. On peut en changer avec les boutons de la barre.
Il est ainsi possible de lancer plusieurs applications par bureau.
- Par exemple on peut travailler sur **Net**scape sur le premier écran, envoyer un message avec le logiciel de messagerie **Kmail** sur le second, chercher un paquetage sur un autre ...
- En bas un panneau escamotable où se trouve à gauche le menu K permettant de lancer les programmes, classés par catégories : applications, Jeux, Graphiques, Internet ... et une série d'icônes pour les principales applications.
A droite, bouton pour la configuration de l'affichage.
En haut, la barre des applications en exécution.
- Sur le bureau, les lanceurs **cdrom** et **floppy** automatisent le montage de ces périphériques, et affichent l'arborescence de leur système de fichiers dans **kfm**
- Les taches d'administration peuvent être effectuées dans un terminal, [Konsole](#)



Manipulations

1. Parcourir la doc en français, manuel utilisateur et FAQ
2. Explorer l'organisation du bureau KDE : panneau, menu K, lanceurs.
3. Lancer quelques applications, par exemple Kmail, Gimp
4. Ouvrir un terminal **Konsole**, y tester quelques commandes courantes : `df`, `cal`, `du`, `date`, `who`, `pstree`, etc ...
5. Lancer des applications en mode console, par exemple **gftp** (absence de lanceur sur le bureau)

Installer un lanceur d'application sur le bureau

- clic-droit sur le bureau/ nouveau /application/ choisir un nom (par ex *KTelnet*)/OK
- clic-droit sur l'icone /propriétés/ onglet *Exécution* / *Parcourir* pour choisir l'exécutable (*/usr/bin/telnet*)
- clic sur l'icone pour choisir une autre icone



Manipulations

Localiser l'exécutable **gftp** à l'aide de **Kfind** (réponse : */usr/bin/*)

Puis installer le lanceur sur le bureau, avec une icone adaptée

Personnalisation du bureau

- Choisir des sons système : **K** /configuration/Sons/Sons du système
- Configurer l'affichage : bouton à droite sur le panneau



Manipulations

- Parcourir les papiers-peints
- Choisir un écran de veille (amateurs d'émotions fortes, visitez "Ecran noir de la mort" !)
- Comment choisir des sons-système ?
- Une petite partie de déminage ? ou de Rubik's cube ?

Le terminal Konsole

```

Midnight Commander
Fichier Sessions Options Aide

Gauche Fichier Commande Options Droite
< /etc > < /home >
Nom Taille MTime Nom Taille MTime
hosts.deny 347 jui 29 1995 /.. 1024 nov 20 10:13
identd.conf 0 jui 12 16:01 /ftp 1024 nov 19 15:28
identd.masq 161 jui 12 16:01 /httpd 1024 nov 19 16:28
im_palette"mall.pal 920 jui 20 10:56 /jean 1024 nov 19 16:28
im_palette-ting.pal 224 jui 20 10:56 /samba 1024 aoû 13 17:28
im_palette.pal 3376 jui 20 10:56 /stage1 1024 nov 19 23:38
inrc 5464 jui 20 10:56
inetd.conf 2999 nov 19 16:26
info-dir 16260 nov 19 16:56
initrunlvl 21 nov 19 16:29
inittab 1756 nov 19 23:58
inputrc 700 sep 16 07:18
ioctl.save 60 nov 20 10:12
irssi.conf 6027 jui 26 23:30
isapnp.gone 1000 sep 14 18:19
issue 1524 nov 20 10:14
issue.net 66 nov 20 10:14
ld.so.cache 45476 nov 19 16:55
ld.so.conf 50 nov 19 16:07
lftp.conf 1076 aoû 18 18:30
lilo.conf 140 nov 19 17:21

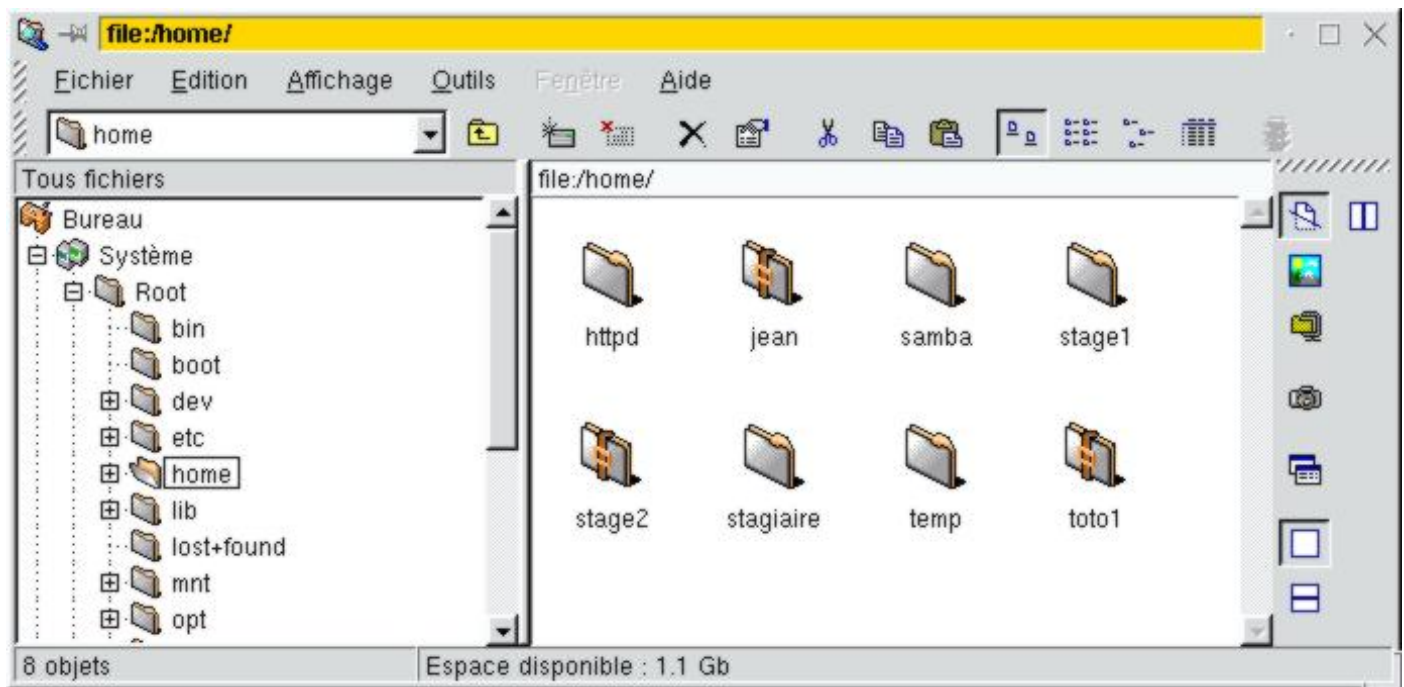
inittab /..

Hint: Tired of these messages? Turn them off from the Options/Layout menu.
[root@p01 /etc]#
1Aide 2Menu 3Voir 4Editer 5Copier 6RenDép 7Mkdir 8Effacer 9Menu de 10Quitter

```

Ce n'est pas qu'un bête terminal, on peut l'utiliser en mode mc !

L'explorateur-navigateur KFM (K-file-manager)



- Pour le lancer, icône *Démarrage* du bureau ou passer par *K / Disk Navigator*
- La commande *Affichage/Arborescence* donne une vue du poste de travail
- C'est un outil servant à la fois de gestionnaire de fichiers local et de navigateur gérant les services d'Internet comme le WEB, FTP.
- En particulier, il interprète tout naturellement le code HTML, et le langage JavaScript.
- Pour passer une requête, saisir l'URL ou utiliser le carnet de *signets*

Applications de KDE

Dans ce qui suit **K** est l'abréviation du menu lié au bouton K situé dans le panneau bas.

Application	Nature	Lancement
konsole	émulateur de terminal	panneau
kedit	éditeur de texte simple	panneau
kwrite	éditeur de texte avancé	
kab	carnet d'adresses	panneau
kmail	client messagerie	panneau
gftp	client ftp	K/Applications X/Internet
knotes	prise de notes	panneau
kcalc	calculatrice	panneau
kfind	utilitaire de recherche de fichiers	panneau
keditmenu	utilitaire de configuration des menus K	K/Utilitaires/Editeur de menus
kruiser	explorateur	panneau/utilities
kfm	navigateur	lanceur Démarrage
ksnapshot	Capture d'écran	panneau/utilities
kview, xv	Visualisateur d'images	panneau/utilities

kghostview	Visualisateur Postscript, pdf	K/Applications
gimp	Traitement d'images	lanceur
rpm	installation programmes	lanceur
linuxconf	configuration système	lanceur
kuser	gestionnaire d'utilisateur	
khexedit	éditeur hexadécimal	
kppp	numéroteur accès distant	K/Internet
klpq	gestionnaire de file d'impression	K/Utilitaires
kpm	gestionnaire de processus	K/Utilitaires
smbstatus	suivi des connexions Samba	K/Configuration/Informations
ksysv	gestion des niveaux d'exécution	K/Système
kDiskFree	indicateur d'espace disque occupé (df)	K/Système

Installation d'applications avec Kpackage

- Il s'agit d'un système de gestion des applications Linux basé sur **RPM** (*RedHat package Manager*)
- Cette interface graphique utilisateur est chargée de :
 - générer les commandes **rpm** pour l'utilisateur.
Pour connaître les principales [options de rpm](#)
 - gérer une base de données des applications installées
- La commande **rpm** est devenue un standard pour faciliter l'installation des programmes . Elle vérifie la dépendance entre les packages et la présence des librairies nécessaires à leur bon fonctionnement.
- syntaxe des packages : `nom.version.i386.rpm`
- droits : opérations réservées à au gestionnaire `root`



Manipulations

- Chercher la liste des fichiers compris dans les packages de `gftp` et de `pine` . Examiner directement leur documentation.
- Sélectionner une applications .rpm dans `kfm`, provoque l'exécution de **KPackage**. Eventuellement procéder à l'installation.
- Monter un cd-rom contenant des applications au format rpm, à partir du lanceur du bureau, ce qui affiche directement son contenu dans le gestionnaire de fichiers `kfm`.
- Installer quelques applications, par exemple l'**éditeur HTML Bluefish**

Installation et prise en main de StarOffice 5.1

(L'installation administrative décrite ci-dessous est peut-être déjà effectuée)



Installation administrative

- Se connecter comme `root`, lancer X-KDE, monter le cd, et dans `kfm`, aller dans `/mnt/cdrom/staroffice/french`
- un simple clic sur le package, lance le gestionnaire **KPackage** en mode installation
- l'installation s'effectue dans `/opt/Office51`



Installation utilisateur

- connexion utilisateur **stagex**
- Dans l'explorateur **Kfm** (icône Démarrage), se placer dans le rép. `/opt/Office51/bin/`, et (simple) clic sur `setup`, lance l'installation utilisateur
(ou si on préfère : dans **Kconsole**, lancer la commande `/opt/Office51/bin/setup`)
- cocher code code média et entrer le code : `xxxx-xxxx-xxxx-xxxx` fourni avec la distribution.
- choisir le type d'installation standard, utilisant l'installation réseau
- les fichiers personnels seront installés dans `/home/stagex/Office51`
- pour lancer staroffice, commande `/home/stagex/Office51/bin/soffice`
- installer un lanceur sur le bureau.

KSnapshott, KView et xv

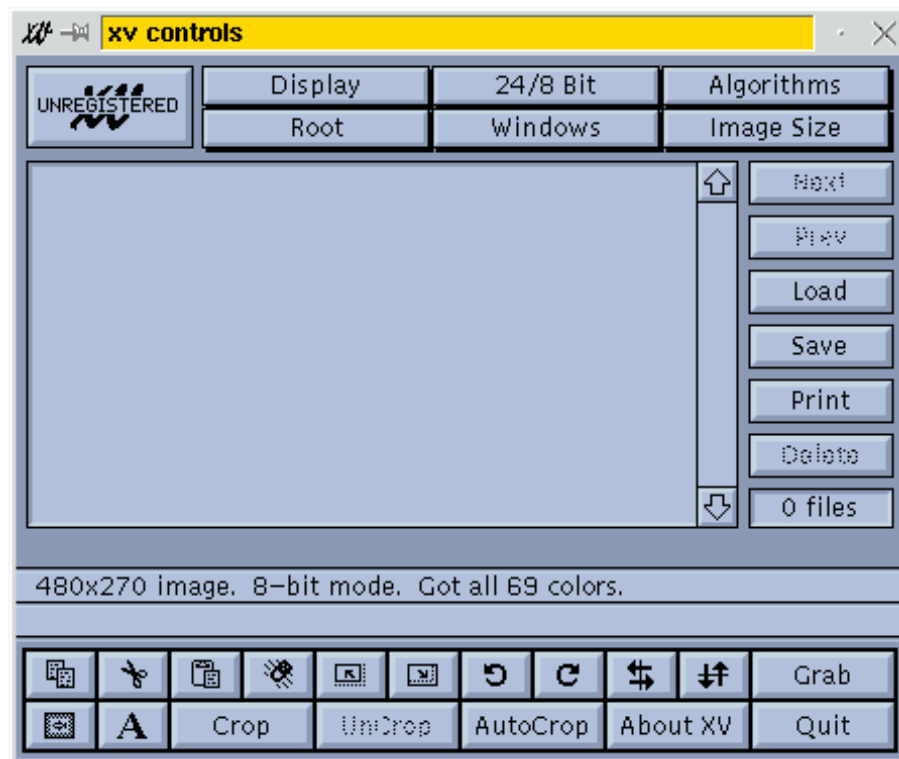
[K/Graphiques/Capture d'écran || Visualisateur d'écran || xv]

Utiliser **KView** pour visualiser des images

xv me semble bien plus facile à utiliser que KSnapshott, notamment pour réaliser les captures d'écrans

Voici une capture de la fenêtre de **xv**, capturée par lui-même !

Les réglages s'effectuent dans la fenêtre **Grab**, ensuite sauvegarder (bouton **Save**)



Lecteur de cd audio kscd

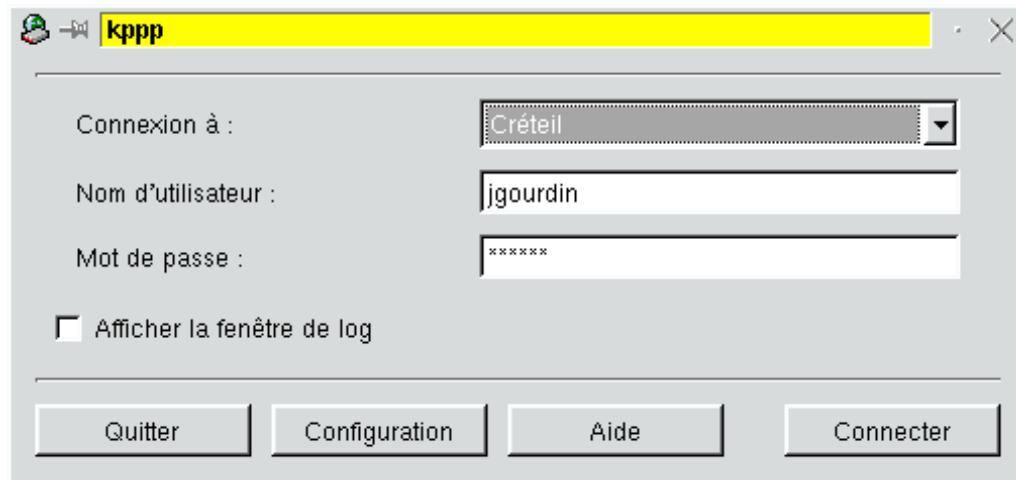
Il faut au préalable installer le module de la carte son. Voir [pour cette installation](#)

Voici une capture, réalisée par **xv** au format jpg, de la fenêtre de **kscd**



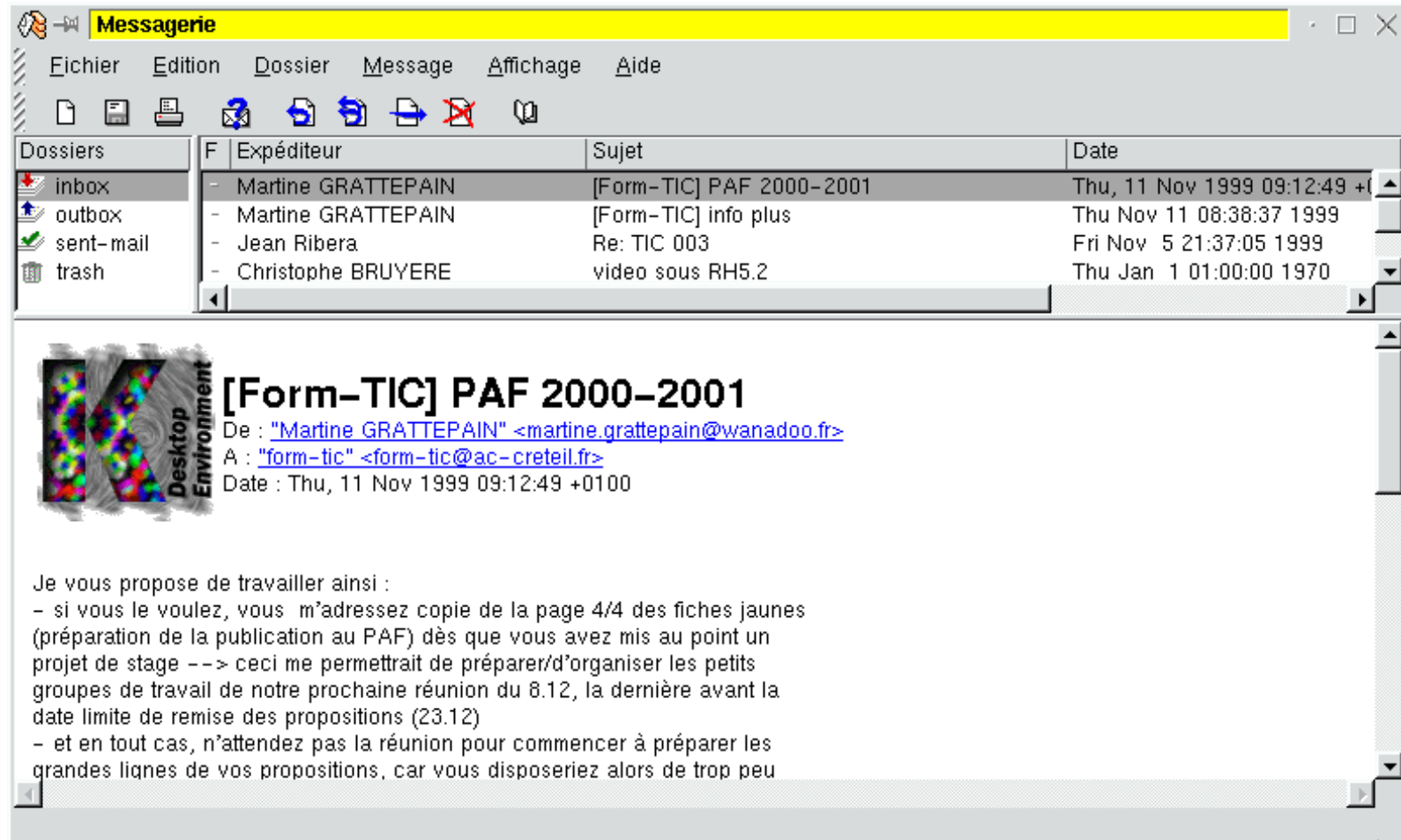
Connexion à Internet par modem (PPP)

Paramétrage aisée avec [Linuxconf](#), si on ne dispose que d'un modem.



Kmail : client de messagerie

Lanceur sur la panneau ou K/Internet/messagerie ou commande kmail dans un terminal, va exécuter le programme /opt/kde/bin/kmail Voir le manuel en français dans le menu **Aide/Contenu**



Fichier /configuration ouvre la fenêtre **Configuration**

Identité

nom : Stagiaire ...

e-mail stagex@ac-creteil.fr
réponse id°
fichier de signature

Réseau

envoi de messages

choisir *par smtp*

serveur : mail.ac-creteil.fr

port : 25

réception de messages

Ajouter / Boîte aux lettres POP3

Nom stagiaire

Utilisateur stagex

Mot de passe *****

Serveur mail.ac-creteil.fr

Port 110

Détruire les messages du serveur ? (cocher si oui)

Charger tous les messages ?

Activer la vérif périodique ?

Possibilité de filtrer : *fichiers/filtrer*



Manipulations

Paramétrer kmail pour consulter votre boîte perso ou à défaut les boîtes stagex@ac-creteil.fr
N'hésitez pas à envoyer largement vos impressions sur Linux.

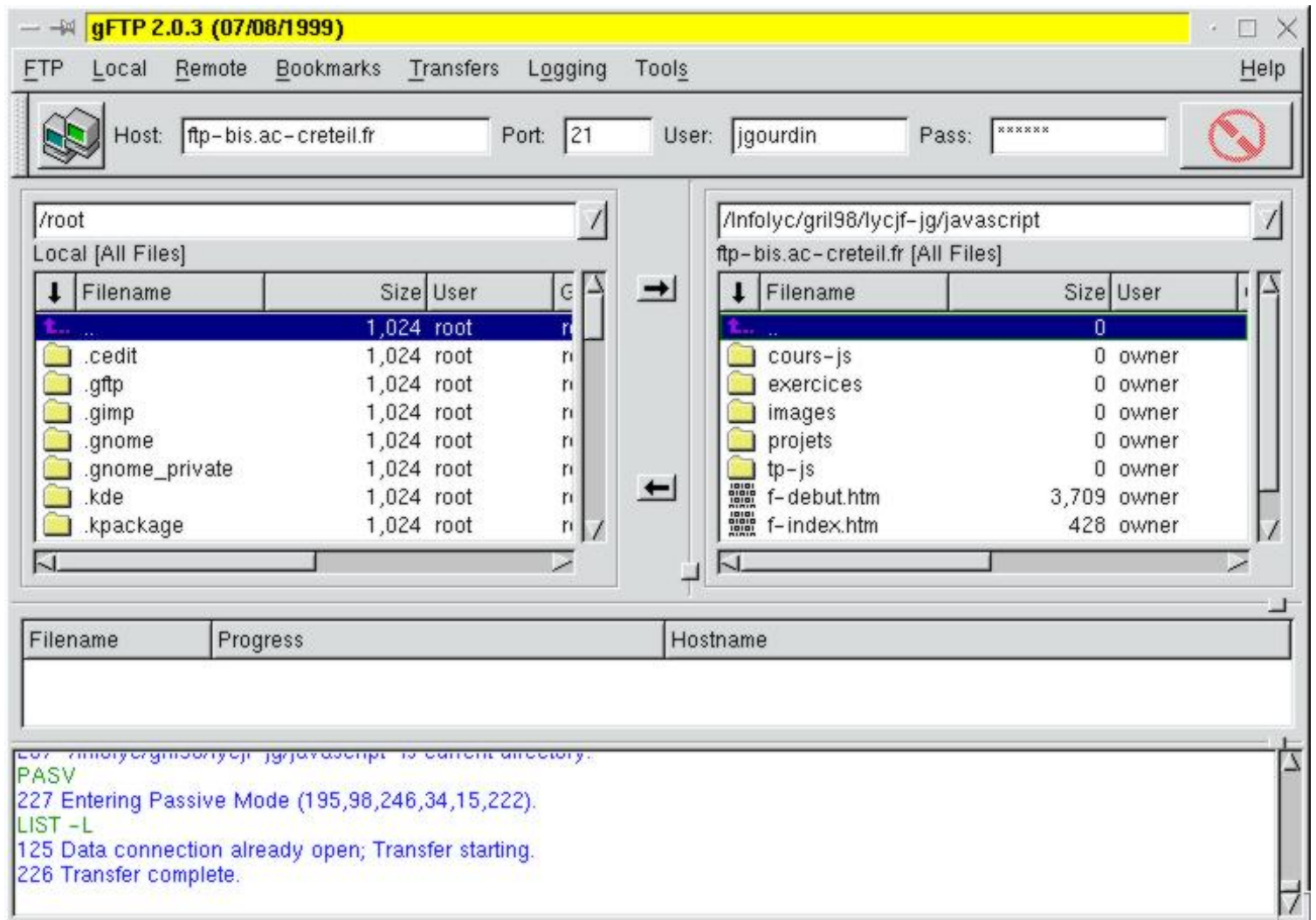
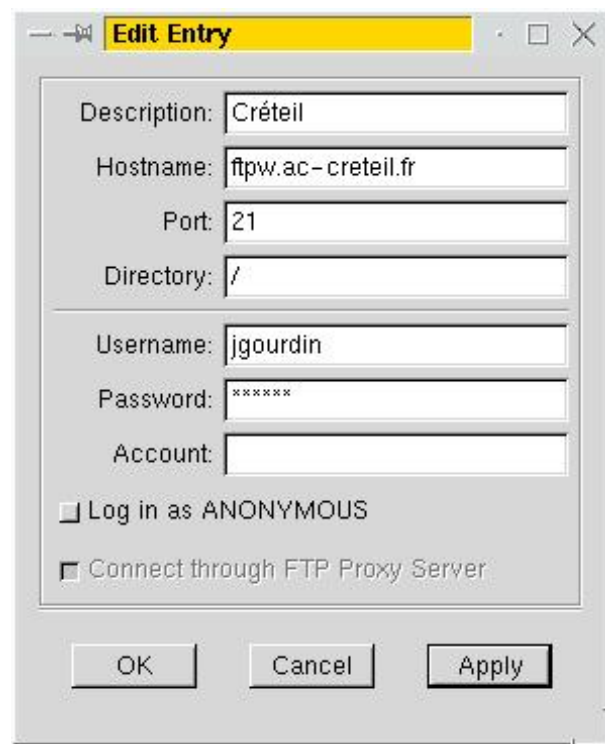
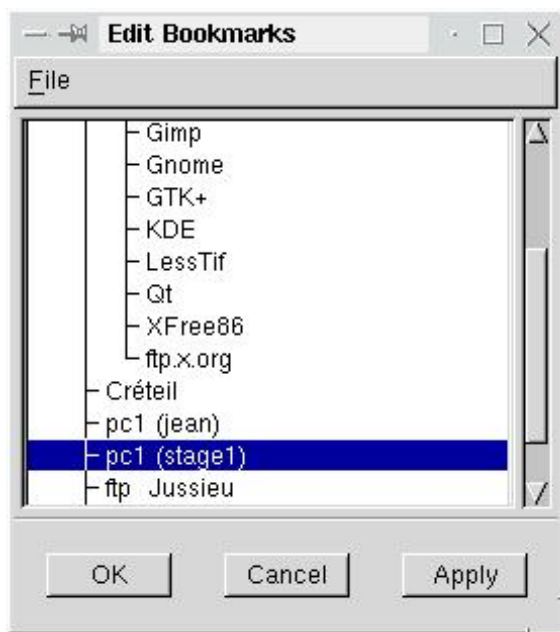
Gftp 2.0.9, client ftp

- C'est un utilitaire très agréable, et ... semblable à WS_FTP32.
- Installer un raccourci sur le bureau, pointant sur /usr/bin/gftp
- Les "*sessions profile*" se paramètrent comme des signets (menu Bookmarks/Edit bookmarks/file/new item..)



Connexion Internet par ftp

Par exemple, voici le paramétrage d'une session (non *anonymous*) sur le ftp de Créteil.



Connexion locale par ftp

- Configurer une connexion FTP avec la station p0y

```
Menu Bookmark / edit/ new folder : p0y
double-clic sur p0y
  Hostname : p0y
  port : 21
  directory : /home/stagey
  username : stagey
  password : stgy
```

- pour connecter : simple clic sur le signet p0y dans Bookmark --> OK
- clic sur /etc/hosts ouvre le fichier dans un petit éditeur
transfert <-- et confirmation de remplacement sur p0x --> OK
- déconnection, modif de la configuration en utilisant le nom de station
double-clic sur p0y
Hostname : p0y
reconnexion --> OK
- tranfert de /etc/bashrc de p0y sur p0x

Gimp, traitement d'images

à découvrir ... et doc à écrire ..

GNOME, l'autre environnement graphique



Découverte de GNOME

Voici un autre environnement de travail graphique, GNOME, respectant la license GPL.

Créé plus récemment encore que KDE, et en plein développement, il est semble t-il préféré à KDE par les "puristes" pour ses possibilités très étendues de personnalisation de l'interface, et ... sûrement aussi parce qu'il ressemble pas à Windows9x !

Toute première approche (sur Mandrake 6.1)

- Sur le compte **root**, rebooter par la commande **reboot**
- Au prompt **LILO** : saisir **linux 5**, pour démarrer directement en niveau graphique.
- Une fenêtre invite à saisir le nom de login et le mot de passe (par exemple, stagex/stgx)
- Dans la liste déroulante *Session Type*, choisir **gnome** et **Go!**
- Pour tester le tableur, lancer **Gnumeric**
- C'est un autre monde, aussi complet que KDE, admirer déjà la variété des écrans de veille !



Installation/paramétrage de XFree86

Avant de la lancer, recueillir le maximum d'informations sur le type de carte vidéo et le type et les fréquences du moniteur.

Au besoin, se servir de Windows !

Installation/modification

Lors de l'installation initiale, le programme exécute l'utilitaire Xconfigurator

Fort heureusement, on peut modifier ce premier paramétrage, sans réinstaller, en lançant comme root :

- soit Xconfigurator
- soit xf86setup (sur certaines distributions)
- Le fichier de configuration est enregistré dans **/etc/X11/XF86Config**
- Si on a sélectionné plusieurs modes, on peut passer de l'une à l'autre avec la combinaison de touches **Ctrl alt +**
- Si le serveur X ne répond plus (le serveur, pas le noyau Linux ;-) on le quitte avec **Ctrl alt BackSpace**
- La doc se trouve à **/usr/X11R6/lib/X11/doc/README.Config**

Démarrage en mode graphique

XFree86 est le nom d'une version libre du serveur X11, l'interface X-Window standard sur Unix

Pour démarrer en mode graphique

- occasionnellement, taper **linux 5**, juste à l'affichage de LILO:
- en permanence, modifier une ligne du fichier
/etc/inittab: id:5: initdefault: (niveau 5 de démarrage par défaut)

Fichiers de démarrage et de configuration

- La commande de connexion au serveur X est **startx**
En fait le fichier exécuté est **/usr/X11R6/bin/startx**.
On peut vérifier que le chemin **/usr/X11R6/bin** figure bien dans la variable **PATH** par la commande **echo \$PATH**.
- Après différents tests, **startx** exécute **xinit**, qui est le véritable lanceur de X. Ce fichier binaire utilise 2 paramètres :
xinit \$clientargs -- \$serverargs
en lisant le code on trouve :
clientargs = /etc/X11/xinit/xinitrc
\$serverargs = /etc/X11/xinit
- Le fichier de configuration **XF86Config**, est généré par le programme **Xconfigurator** et décrit toutes les caractéristiques matérielles du matériel (carte, moniteur ...).
Il se trouve physiquement à **/etc/X11/XF86Config**
- Lorsque le serveur XFree86 est lancé, il cherche un **Window Manager** (système de fenêtrage et de gestion de bureau). Si l'utilisateur n'a pas défini de configuration personnalisée, la config se poursuit de façon

générale par l'exécution du script **/etc/X11/xinit/xinitrc**

- Pour comprendre mieux les mécanismes, [voir les annexes](#)

Personnaliser X et le Window Manager

Pour personnaliser l'interface graphique, choisir un **Window Manager**, gestionnaire de fenêtre personnel :

- copier le fichier général **/etc/X11/xinit/xinitrc** dans le rép. personnel \$HOME
- le modifier (voir man xinitrc)

Exemple :

```
] $ less $HOME/.xinitrc

# WindowMaker Default

exec /usr/X11R6/bin/wmaker
```

En cas de plantage

- Pour sortir de X, repasser dans la console texte, où a été lancé **startx**, par la combinaison de touche **Ctrl-Alt-Fx**, où x est le numéro de cette console.
Ceci a pour effet de "tuer" le serveur X lui-même, et donc de faire repasser l'utilisateur en mode texte.
- En cas de problème, presser la combinaison de touches **ctrl-alt-backspace** en même temps.
- Une fois sous X, on peut passer d'une console à une autre grâce à la combinaison de touche **ctrl-alt-Fx** où x est une valeur de 1 à 6 pour les consoles textes .
- Pour comprendre un problème, il faut examiner le contenu du fichier **/tmp/x.out**, qui contient les messages d'erreurs et les avertissements.

Annexes

Voici quelques extraits de scripts commentés (Mandrake 6.1)

Tous les exécutables X se trouvent dans **/usr/X11R6/bin**

notamment le script **startx**

```
-----
#!/bin/sh
# (c) 1999 Red Hat Software, Inc., MandrakeSoft
# chemin
bindir=/usr/X11R6/bin

userclientrc=$HOME/.xinitrc
userserverrc=$HOME/.xserverrc
sysclientrc=/usr/X11R6/lib/X11/xinit/xinitrc
sysserverrc=/usr/X11R6/lib/X11/xinit/xserverrc
sysclientrc=/etc/X11/xinit/xinitrc
sysserverrc=/etc/X11/xinit/xserverrc
clientargs=""
```

```

serverargs=""
# la variable clientargs désigne :
# le fichier .xinitrc s'il existe dans le rép. perso
# le fichier etc/X11/xinit/xinitrc sinon
if [ -f $userclientrc ]; then
    clientargs=$userclientrc
else if [ -f $sysclientrc ]; then
    clientargs=$sysclientrc
fi
fi

# de même la variable serverargs désigne ici /etc/X11/xinit/xserverrc
if [ -f $userserverrc ]; then
    serverargs=$userserverrc
else if [ -f $sysserverrc ]; then
    serverargs=$sysserverrc
fi
fi
.....

# appel du script xinit
xinit $clientargs -- $serverargs
-----
# extrait du fichier /etc/X11/xinit
# l'utilisateur peut avoir son propre client graphique $HOME/.Xclients
# par défaut, c'est le cas ici, on lance /etc/X11/xinit/Xclients

if [ -f $HOME/.Xclients ]; then
    exec $HOME/.Xclients
elif [ -f /etc/X11/xinit/Xclients ]; then
    exec /etc/X11/xinit/Xclients
else
    # failsafe settings. Although we should never get here
    # (we provide fallbacks in Xclients as well) it can't hurt.
    xclock -geometry 100x100-5+5 &
    xterm -geometry 80x50-50+150 &
    if [ -f /usr/bin/netscape -a -f /usr/doc/HTML/index.html ]; then
        netscape /usr/doc/HTML/index.html &
    fi
    if [ -f /usr/X11R6/bin/fvwm ]; then
        exec fvwm
    else
        exec twm
    fi
fi
-----
# xinitrc lance à son tour Xclients
# voici ub extrait de /etc/X11/xinit/Xclients

```

```

# on cherche si l'utilisateur a un bureau préféré
PREFERRED=
# si le fichier desktop existe dans /etc/sysconfig
if [ -f /etc/sysconfig/desktop ]; then
    if [ -n "`grep -i GNOME /etc/sysconfig/desktop`" ]; then
        PREFERRED=gnome-session
    elif [ -n "`grep -i KDE /etc/sysconfig/desktop`" ]; then
        PREFERRED=startkde
    elif [ -n "`grep -i AnotherLevel /etc/sysconfig/desktop`" ]; then
        PREFERRED=AnotherLevel
    fi
fi
#
if [ -n "$PREFERRED" -a "$PREFERRED" != "AnotherLevel" ] && \
    which $PREFERRED >/dev/null 2>&1; then
    PREFERRED=`which $PREFERRED`
    exec $PREFERRED
fi

# si on arrive jusqu'ici, c'est qu'on veut AnotherLevel ou
# il n'y a pas de fichier desktop et la variable PREFERRED n'est pas définie
# si la chaîne "$PREFERRED" est restée vide
if [ -z "$PREFERRED" ]; then

    GSESSION=gnome-session
    STARTKDE=startkde

# par défaut, on lance KDE
    if which $STARTKDE >/dev/null 2>&1; then
# pour cela, on exécute la commande /usr/bin/startkde
        exec `which $STARTKDE`
    fi

    # if KDE isn't installed, try GNOME
    if which $GSESSION >/dev/null 2>&1; then
# on exécute la commande /usr/bin/gnome-session
        exec `which $GSESSION`
    fi
fi

```



Installation LINUX

Distribution : *Mandrake 6.1*

Attention ! Il s'agit de la description d'une installation particulière, sur les machines du Centre de Formation de l'académie de Créteil (CFIPEN)

Installation initiale

Préalable

Bon, Windows peut nous rendre encore quelques services ...

Relever dans son panneau de configuration/système les caractéristiques des périphériques et cartes diverses installées sur la machine : vidéo, son, réseau .. les irq et adresses io qu'elles utilisent.

Ainsi, sur les machines des salles du CFIPEN, on observe pour

* la carte réseau D-Link TX 530 : irq= , adresse io=

* la carte vidéo ATI "RagePro" * la carte son compatible Sound Blaster : irq= , port i/o= , canaux DMA 1= et DMA 2=

La disquette d'installation est nécessaire si le bios des machines ne permet pas de booter directement sur le cd Linux.

Si la distribution ne la fournit pas, il faut la préparer :

Se mettre sous DOS et lancer les commandes suivantes :

C:\> [lecteur-CD]:\dosutils\rawrite

source : [lecteur-CD]:\images\boot.img

cible : A:

Procédure séquentielle d'installation

- Booter sur la disquette d'installation
- Installation ou mise à jour : Enter
- choix de la langue d'installation : French
- config clavier : fr-latin1
- installation par CD-ROM ou par disque dur
- installation ou mise à jour
- choix du type d'installation : station, serveur ou personnalisée
- présence d'adaptateur SCSI : non
- Création des partitions du disque, préférence pour fdisk

Il est recommandé de créer une partition de swap de 128M pour une RAM de capacité inf à 64M (sinon 80M suffit).

Il faut au moins une partition Linux. Mais il est recommandé d'en créer une spécialement pour /home, pour loger les rép. personnels.

Voir d'[autres choix de partitions](#).

- partitions actuelles (dépend bien sûr de l'état du disque) à supprimer
- principales commandes : m help, donne la liste complète des commandes
p pour afficher les partitions courantes, d supprimer une partition, n en ajouter
- d delete , supprimer toutes les partitions existantes
- n ajout -- p primary partition -- 1 (1ère partition)
first cylindre 1 -- last +2000M (par exemple)
- p --> /tmp/hda1 1 255 83 Linux native
- n ajout -- p primary partition -- 2 (2ème partition)

first cylindre 256 -- last +128M

- **p** --> /tmp/hda2 256 272 83 Linux native
- **t** pour changer de système pour la partition 2
 - Numéro de la partition 2
 - Code hexa 82 (swap)
- **w** --> écriture de la table et sortie / Fait
- affichage de la table des partitions :
édition ligne hda1, ajout de / pour indiquer le point de montage racine du système de fichiers.

/	hda1	2000	2000	Linux native
	hda2	133	133	Linux swap

hda	[525/255/63]	4118	2133	1985	free
-----	--------------	------	------	------	------

- formatage de l'espace de swap sur /tmp/hda2
- formatage de /dev/hda1 / avec vérification des blocs défectueux pendant le formatage
- choix des paquetages à installer parmi la liste des groupes de paquetages
pour choisir plus finement, cocher sélection individuelle des paquetages
choix conseillé :
 - printer
 - X Window system
 - KDE
 - GNOME
 - Office extensions
 - Mail/WWW/News tools
 - Dos/Windows connctivity
 - Console Multimédia
 - File Managers
 - Graphics manipulation
 - Console multimédia
 - X multimédia support
 - Networked WorkStation
 - NFS server
 - SMB
 - WEB
 - DNS
 - network management Workstation
- Trace de l'installation dans /tmp/install.log
- création d'un système de fichiers ext2 sur /dev/hda1 ..
puis installation des 419 paquetages sélectionnés, au total 507M, durée environ 35 minutes(10 mn maintenant)
- Probing found some type of serial mouse on port ttyS0 : auto-détection de la souris série sur le port com 1
souris générique et émuler le 3ème bouton
- configuration réseau oui
La carte réseau est reconnue ! --> carte VIA Rhine (pour la carte réseau D-link 530 TX, à lier à l'interface **eth0**, indiquer le module **via-rhine**, sans préciser d'irq , ni d'adresse io.
On vérifiera après le processus d'installation que la carte est bien installée avec la commande
ifconfig eth0
- configurer les zones horaires : Europe/Paris

- services à démarrer [automatiquement lors du boot](#)
Chacun peut être choisi après examen d'une fenêtre d'aide (F1. On peut par exemple ajouter nfs, rstatd et enlever pcmcia, sendmail.
 - Choix de connexion de l'imprimante : locale, lpd distant, SMB/Windows 95/NT, NetWare
 - Installation différée : on peut ensuite lancer l'utilitaire graphique **nettool**
 - Si on installe l'imprimante, voici un exemple de configuration
 - nom de la file : lp
 - rép de spool : /var/spool/lpd/lp
 - périphérique imprimante /dev/lp0 (l'équivalent de LPT1:)
 - modèle : HP Desjet 500
 - taille : a4
 - correction de l'effet d'escalier du texte
 - choix du mot de passe de root : **cfipen** (2 fois)
 - ajout d'un autre utilisateur : nom : stage1 / mot de passe : stg1/shell : Bash / OK
 - authentication configuration
 - [] Enable NIS
 - [*] Use Shadows Passwords
 - [*] Enable MD5 Passwords
 - création d'une dk de démarrage : très conseillé !
 - installation de LILO :
chargeur de démarrage dans /dev/hda MBR ou /dev/hda1 1er secteur de la partition de démarrage
 - exécution de **Xconfigurator** 4.2.3
D'abord quelques remarques
On peut lancer cet utilitaire de configuration du serveur X n'importe quand, bien sûr connecté comme root.
Il reconnaît les cartes vidéos les plus courantes.
La difficulté (actuelle) vient plutôt du paramétrage du moniteur.
Si la configuration acceptée ou choisie (résolution et nombre de couleurs) nous conduit à un plantage du serveur X, ce n'est pas dramatique, pas besoin de redémarrer ;-)
--> Ctrl-alt-Fx où x est le numéro de terminal, puis Ctrl-C, et on relance Xconfigurator, en choisissant une autre config. Déroulement
La carte graphique (ATI "RagePro") est reconnue ! Match 64 GB
Le serveur X installé est : Match 64
choix du moniteur Samsung SyncMaster 15GL dans la liste !
Tester
Choix des modes vidéo par défaut 800x600 , 8 bits par pixels
Choix de démarrer en mode graphique : non
 - reboot sur le disque
Première connexion en mode 3 (texte) puis en mode 5 (X-KDE) --> OK
-

Installation de paquetages rpm

On peut toujours installer "à la main" des applications qui n'auraient pas été choisies lors de l'installation initiale. Voici la procédure sur l'exemple d'installation de Midnight Commander.

- mount /dev/cdrom --> "monter" le cdrom
- cd /mnt/cdrom/Mandrake/RPMS --> se positionner dans le point de montage
- rpm -vih mc-... --> compléter le nom du fichier avec la touche TAB, et valider

- `mc -->` pour lancer l'utilitaire
 - `whereis mc -->` pour savoir où l'exécutable a été installé (`/user/bin/mc`)
-

Installation mixte en dual-boot

- Ne pas installer tout d'abord Linux
Sur un disque Linux déjà installé, connexion root, puis `fdisk /dev/hda --->` suppression de toutes les partitions
Tentative puis échec de partitionner le disque sous Linux puis d'installer Linux.
Refus d'installer LILO dans le MBR
Ensuite Windows ne reconnaît pas le type de système de fichiers affecté à `/tmp/hda1` (Win95 FAT32)
 - Partitionnement DOS et formatage de la partition principale
Boot sur `dk dos`, lancement `fdisk dos`
ajout d'une partition principale DOS de 200 Mo (25%)
partition 1 DOS activé
reboot
attribution du lecteur C: à la partition 1
formatage de C: --> `format c: /s`
version du DOS installé : `ver -->` Windows 95 (4.00.1111)
remarque : lilo, déjà présent dans le MBR, n'a pas été détruit ! et empêche d'accéder à C:
suppression de lilo ---> `a:\fdisk /mbr`
 - Installation Linux
`choix fdisk`
`/tmp/hda1 * 1 204 M Dos 16-bit >=32`
`/tmp/hda2 Extended`

suppression de la partition étendue
ajout des partitions swap et linux en `hda2` et `hda3`
Installation de lilo dans le MBR
Reboot sur Linux --> ok !
 - Installation Windows
Reboot sur DOS : accès à C:

`install` à partir de la `dk dos`, du pilote `cdrom`

`E:\> install.exe` pour installer Win95 en mode compact

redémarrage : lilo n'est pas utilisé !!!
Probablement effacé du MBR par Windows?
`r>`
 - Réinstallation de LILO
Boot sur `dk linux` de démarrage
Re-install de lilo dans le MBR par la commande :
`# lilo` avec option `dos` par défaut
reboot avec succès sur chacun des 2 systèmes !
-

Utilitaires de configuration

Il peut être indispensable de réinstaller des périphériques, par exemple lors d'un déplacement de port série d'une souris, d'ajout d'une carte son, de changement de moniteur ...

Si on peut accéder à un serveur X, utiliser alors les utilitaires graphiques plus conviviaux.

- *mouseconfig paramétrage souris*
 - *kbdconfig clavier*
 - *Xconfigurator carte vidéo et moniteur*
 - *sndconfig carte son*
 - *ntsysv services à démarrer automatiquement*
 - *mkbootdisk --device /dev/fd0 2.2.13-7mdk créer une disquette de démarrage*
 - *printtool installation d'une imprimante en mode graphique*
-

Configuration réseau

- Dans le cas où la carte réseau n'est pas reconnue lors de l'installation initiale par la distribution, il faut procéder à son installation "manuelle".
Sous windows, noter son type, son irq et son adresse io
- Il faut ensuite chercher un module générique susceptible de reconnaître cette carte !
Par exemple, pour une carte ISA assez ancienne, identifiée sous Windows comme une SMC Ethernet plus Elite 16 (WD/8013W), il faut utiliser le module `wd.o` ! pas évident !
- On peut renseigner directement le fichier `/etc/conf.modules` de configuration avec un éditeur (par exemple `mc`), mais ce travail dispersé est avantageusement remplacé par l'utilitaire **linuxconf** (ici utilisé en mode texte)

Voici la description des 2 possibilités :

1ère solution, en ligne de commande

`/etc/conf.modules` doit contenir la liste des modules chargés par le noyau, et en particulier le pilote de la carte réseau ISA.

Ajouter (ou remplacer) les lignes suivantes :

```
alias eth0 wd # assure la liaison de la 1ère interface réseau au pilote wd.o
# (pour carte D-Link TX 513)
options via-rhine # les paramètres io et irq sont facultatifs
```

Il est inutile de rebooter !! Quel étonnement ;-)

Passer la commande `/etc/rc.d/init.d/inet restart` pour relancer les fonctions réseaux.

Le diagnostic de fonctionnement l'interface réseau Ethernet **eth0**, est obtenu par la commande **ifconfig eth0**

Ne pas hésiter aussi à "pinguer" les machines voisines.

Ensuite, éditer les [fichiers réseaux](#) pour paramétrer TCP/IP et les services réseaux. Voici les fichiers de configuration à mettre à jour

`/etc/sysconfig/network-scripts/ifcfg-eth0` *fichier de configuration de l'interface eth0*

```
DEVICE = eth0
IPADDR = 10.194.2.100 + x (x étant le numéro de poste)
NETMASK = 255.255.255.0
NETWORK = 10.194.2.0
BROADCAST = 10.194.2.255
ONBOOT = yes
```

`/etc/sysconfig/network`

```
NETWORKING = yes
HOSTNAME = p0x.cfipen.fr
DOMAINNAME = cfipen.fr
```

```
GATEWAY = 10.194.2.245    passerelle par défaut du réseau local
GATEWAYDEV = eth0
NISDOMAIN=""
```

```
/etc/host.conf
order hosts, dns
multi on
```

```
/etc/resolv.conf
#nom de domaine local de l'ordinateur
domain cfipen.fr
# adresse du serveur primaire DNS de Créteil
nameserver 195.98.246.50
# liste de domaines à essayer, si le nom d'hôte ne précise pas son domaine
domainsearch ac-creteil.fr
```

/etc/networks inutile ici, non créé.

Pour tester la "visibilité" des différentes machines, "pinguer" par la commande :
ping 10.194.2.100+y

2ème solution avec linuxconf

[root@p0x /] linuxconf

menu : Configuration / Réseau / Tâches clientes

Config de base de la machine

Nom de machine	p00	p0x
Adaptateur 1 activé	<u>config manuelle</u>	Dhcp bootp
nom complet	p00.cfipen.fr	p0x.cfipen.fr
alias	p00	p0x
adresse IP	10.194.2.100	10.194.2.100+x
masque	255.255.255.0	
interface réseau	eth0	
module noyau	via-rhine	
port		
irq	(optionnel, donc peut être récupéré)	

laisser (les adaptateurs 2 à 4 libres)

Résolution des noms (DNS)

Usage DNS	à cocher pour une connexion Internet
domaine par défaut	cfipen.fr
nom de domaine 1	195.98.246.50 ! ici <u>numéro IP</u> du DNS-provider
nom de domaine 2 , 3	
domaine de recherche	ac-creteil.fr

Routage et passerelles

Passerelle par défaut	10.194.2.245
	(adresse IP routeur, pour PPP ne rien mettre)
Activer le routage [x]	
Autres routes	<i>pas de passerelles vers d'autres sous-réseaux locaux</i>
le démon de routage	<i>désactivé, n'exporte aucune route</i>

Chemin de recherche pour le nom de machine

Adresses multiples pour une machine [x] ??

ordre de recherche (o) hosts, dns *recherche locale d'abord puis internet*

Validation

"Voir ce qui doit être fait" --> il est prévu d'exécuter les processus suivants :

```
/etc/rc.d/rc3.d/S05apmd start
/etc/rc.d/rc3.d/S10network reload
/etc/rc.d/rc3.d/S50 inet restart
/etc/rc.d/rc3.d/S85gpm start
/etc/rc.d/rc3.d/S85httpd start
/etc/rc.d/rc3.d/S90xfs restart
```

Il est inutile de rebooter !! Quel étonnement ;-)

Dans l'écran d'accueil, on voit avec satisfaction, l'identification réseau de la machine :

p0x.cfipen.fr

conseil Même si Linuxconf fait très bien pour nous le paramétrage, il est instructif d'examiner les fichiers de configuration

Essai Connexion réseau local :

interrogation des autres machines par **ping**, session ftp ou telnet

Connexion Internet :

passer sous X, dans kfm ou Netscape, lancer des requêtes HTTP vers www.linux-mandrake.com/fr/,
www.ac-creteil.fr, par exemple ...

Compléments

Config de la carte son

Cette installation, non prévue dans le processus d'installation, est facilement effectuée avec l'utilitaire

/usr/sbin/sndconfig qui détecte la présence d'une *Creative SB16 PNP*.

lancer la commande dans un terminal Kconsole (en mode X)

la carte SB est reconnue, mais encore problème de configuration au cfipen
(l'irq 5 étant occupé par ... la carte réseau)

on peut entrer aisément les paramètres de la carte :

port e/s	irq	dma 1	dma 2	MPU E/S
0x220	10	3	5	0x300

essai, audition médiocre d'un message de bienvenue --> ça fonctionne !

Examiner ce que l'installation a ajouté dans `/etc/conf.modules`

Et bien sûr, passer sous KDE, placer un CD audio et lancer l'utilitaire **kscd** (icône dans le panneau bas, à droite)

Problème de droit parfois rencontré: il faut augmenter les droits sur le rep spécial `/dev/cdrom` par `chmod 666 /dev/cdrom` (il n'y aurait pas de processus de montage pour lire les cd audio ?)

Installation lecteur ZIP sur port parallèle

- `modprobe -c` liste les modules
- `modprobe ppa` *installe le module ppa.o*
ppa version 2.03
ppa : found device at id6, attempting to use SPP
ppa : communication established with id6, using SPP
scsi0 : iomega VP10 (ppa interface)
.... Detected scsi removable disk sda at scsi0, channel 0, id 6, lun 0

- `mkdir /mnt/zip` *création du rép. de montage*
- `mount /dev/sda1 /mnt/zip` *essai de montage d'une dk zip avec le fichier spécial sda1*
sda : sda4
mount : /dev/sda1 is not a valid block device
- `mount /dev/sda4 /mnt/zip` --> ok
A noter comme pour un cd-rom monté, que le bouton d'éjection est inhibé.
- Il reste maintenant à intégrer la chargement du module lors du démarrage
Pour cela, on peut :
éditer le fichier **/etc/rc.d/rc.local**
ajouter à la fin la commande **modprobe ppa** Il ne reste qu'à monter une dk zip, avec la commande `mount` .
- Si on travaille sous KDE, il est judicieux de poser sur le bureau une icone de montage, comme celles du cd-rom et du floppy.
On procède comme pour installer un lanceur, *clic-droit/Nouveau/Périphérique système de fichiers*
- Problème rencontré.
Le lecteur n'est plus monté. La commande **dmesg** fournit les commentaires de démarrage.
Le lecteur s'est installé dans le device `sda1` (pourquoi `sda1` ou `sda4` ?).
La table de montage `fstab` n'est plus à jour, il faut alors `mount(er)` à la main.

Configuration d'une connexion PPP

modem installé sur Com2

Menu *K/Internet/numéroteur*

Dans la fenêtre **kppp**, bouton *Configuration*

Dans la fenêtre *Configuration de kppp*,

onglet Comptes / bouton *Nouveau* ..

Dans la fenêtre *Nouveau compte*

onglet *Numérotation* :

donner un nom à la connexion (Wanadoo), n° de téléphone
et cocher mot de passe.

onglet *IP* :

laisser adresse IP dynamique

onglet *DNS* :

entrer le nom de domaine du fournisseur d'accès, et
les adresses IP des serveurs de noms.

onglet *Passerelle* :

par défaut

assigner l'itinéraire ...

OK

onglet Périphériques paramétrage du modem

choix du périphérique spécial (`com2` --> `/dev/cua1`)

vitesse 57600

Utiliser un fichier de verrouillage

onglet Modem interroger le modem

Essai connexion :

Nom :

Mot de passe :

WEB : essai avec KFM puis avec Netscape

FTP : connexion à `ftpw.ac-creteil.fr`

messagerie

Lanceur du numéroteur

clic-droit/Nouveau/application/kppp.lnk
parcourir/ouvrir /usr/bin/kppp

Utilitaires de configuration

Il peut être indispensable de réinstaller des périphériques, par exemple lors d'un déplacement de port série d'une souris, d'ajout d'une carte son, de changement de moniteur ...

Si on peut accéder à un serveur X, utiliser alors les utilitaires graphiques plus conviviaux.

- mouseconfig *paramétrage souris*
- kbdconfig *clavier*
- Xconfigurator *carte vidéo et moniteur*
- sndconfig *carte son*
- ntsysv *services à démarrer automatiquement*
- mkbootdisk --device /dev/fd0 2.2.13-7mdk *créer une disquette de démarrage*
- printtool *installation d'une imprimante en mode graphique*

Remarques

1. préférence pour fdisk :
j'ai eu des pbs avec Disk Druid, avec la reconnaissance de partition étendue DOS
2. Exemple de partitionnement d'un disque de 6 Go La partition 1, créée d'abord avec fdisk DOS et déjà formatée, est destinée à recevoir une installation Windows 9x
Les partitions 2 et 3 sont affectées à la racine / du système de fichiers Linux et à la zone d'échange (swap).
La partition 4 est de type *étendue* pour pouvoir ensuite y créer 2 partitions "logiques", numérotées 5 et 6. Celles-ci sont destinées respectivement à recevoir les répertoires personnels **/home** et les points de montage des périphériques **/mnt**

Mount	Point	Device	Requested	type
		hda1	1000	Vfat
/		hda2	2000	linux 83
swap		hda3	125	Linux 82
/home		hda5	1004	Linux
/mnt		hda6	100	Linux

3. automatiquement lors du boot :
On peut changer les services à lancer au boot, avec l'utilitaire **ntsysv** ou dans un des menus de **linuxconf**



Outils de configuration et d'administration

Utilisation de telnet

Telnet permet de prendre le contrôle d'un serveur à distance, il y a donc danger potentiel.

Par sécurité, le module serveur n'est plus installé par défaut sur certaines distributions.

Pour l'installer, monter le cd installation et installer le package `telnet-serveur...rpm`

Normalement root ne peut pas se connecter par telnet. Pour le permettre, mettre en commentaire (avec un #) la ligne de `/etc/pam.d`

```
#auth required /lib/security/pam_securetty.so
```

LINUXCONF en mode texte (v 1.16)

Cet utilitaire n'est qu'un programme frontal qui écrit ou met à jour les fichiers de configuration de `/etc`

Mais quel progrès ! il permet enfin d'administrer de façon centralisée, sans se perdre dans les divers fichiers ...

Et `Linuxconf` active immédiatement les services mis en place, relance les démons ...

Naturellement, il ne dispense pas de connaître les actions effectuées.

Ses fichiers, notamment d'aide se trouvent dans `/usr/lib/linuxconf`

Root seul peut lancer `linuxconf` (`ounetconf` qui va directement dans la configuration réseau).

En session X-KDE, un utilisateur "quelconque" qui active le lanceur `linuxconf` se voit proposer de passer root (avec la commande `su`)

Consulter les fichiers d'aide `intro.help`, `netconf.help`, etc.. accessibles avec les boutons *Aide*. Ils contiennent l'essentiel des infos sur les commandes.

Comme d'habitude en mode texte, utiliser `TAB` pour passer d'un champ au suivant.

Pour certains champs, signalé par une flèche vers le bas, `Ctrl-X` ouvre une liste déroulante dans laquelle on peut choisir.

Le menu principal offre 2 types d'actions : *Configuration* et *Contrôler*

Configuration Réseau

Config de base de la machine

Nom de machine	pxx	
Adaptateur 1 activé	config manuelle	
nom complet	pxx.cfipen.fr	pcx.maison.fr
alias	pxx	pcx
adresse IP	10.194.2.(100+xx)	192.168.1.245
masque	255.255.255.0	
interface réseau	eth0	
module noyau	wd	
port	0x240	0x300
irq		5 ou 12

Adaptateurs 2 à 4 libres

Résolution des noms (DNS)

Usage DNS	à cocher pour une connexion Internet		
domaine par défaut	cfipen.fr	maison.fr	
nom de domaine 1	195.98.246.50	193.252.19.3	DNS du provider
nom de domaine 2		193.252.19.4	
domaine de recherche	ac-creteil.fr	wanadoo.fr	

Routage et passerelles

Passerelle par défaut	10.194.2.245	192.168.1.245	IP du routeur/ISB100
autres routes	pas de passerelles vers d'autres sous-réseaux locaux		
le démon de routage	désactivé, n'exporte aucune route		

Chemin de recherche pour le nom de machine

Adresses multiples pour une machine ??

PPP/SLIP/PLIP

Ajout PPP

Tâches serveur

Systèmes de fichiers NFS

DNS

Serveur WEB Apache

Comptes utilisateurs

Les commandes sont évidentes, correspondent à ce qui a été vu en mode ligne de commande.

Le grand intérêt est de permettre de gérer les droits en même temps que les comptes et les groupes.

Création d'un nouveau compte Donner seulement : nom de login et nom complet, par exemple (stagex / stgx)

A la validation, on observe que Linuxconf exécute les 2 commandes suivantes :

```
/usr/sbin/useradd -m -c 'nom' -d /home/login -s '/bin/bash' -G '' -e '' login  
/usr/bin/chage -m -1 -M 99999 -W -1 login
```

Puis on est averti que le mot de passe a bien été enregistré dans `passwd` et `smbpasswd`

Remarquer le rép. personnel créé avec le mode de permissions 0700

Définition des groupes

On peut créer de nouveaux groupes, par exemple `stagiaire`

Il est facile d'affecter des utilisateurs aux groupes (dans le champ autres membres, écrire la liste avec des espaces uniquement).

Systèmes de fichiers

Accéder au disque local

Services divers

Choix du niveau d'initialisation du système : 3 par défaut et 5 en mode graphique

mode de démarrage

Paramétrage de LILO

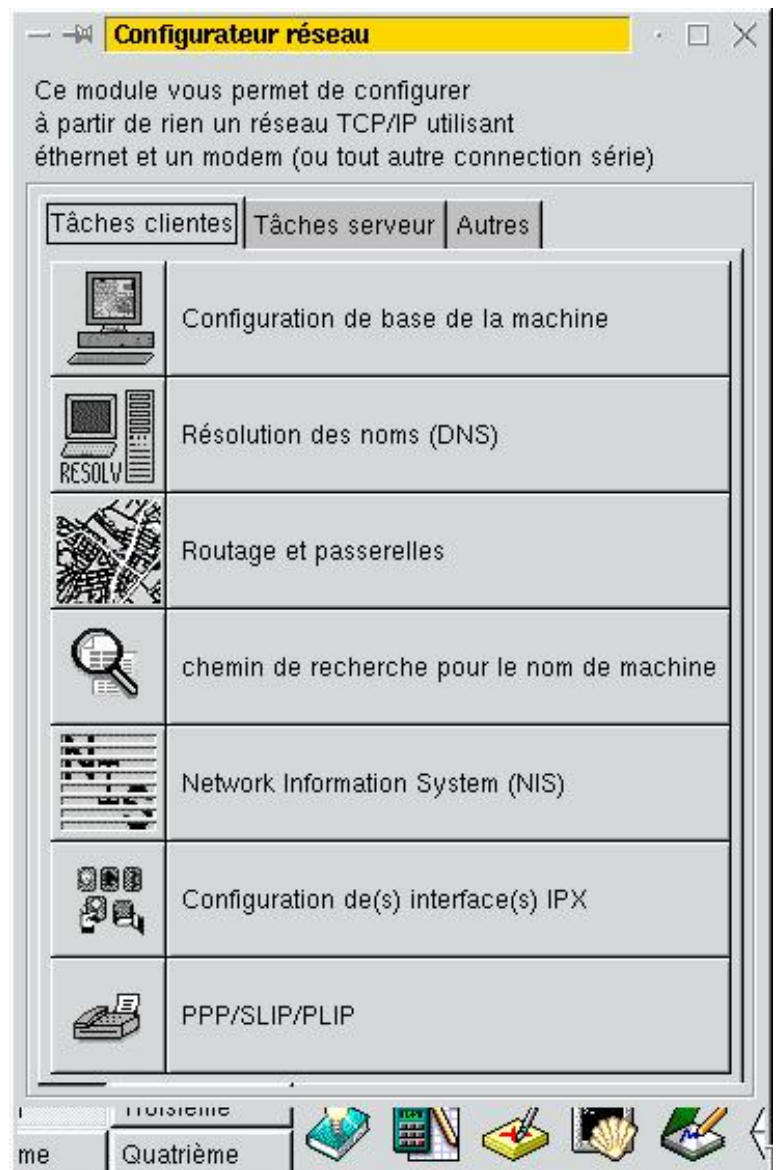
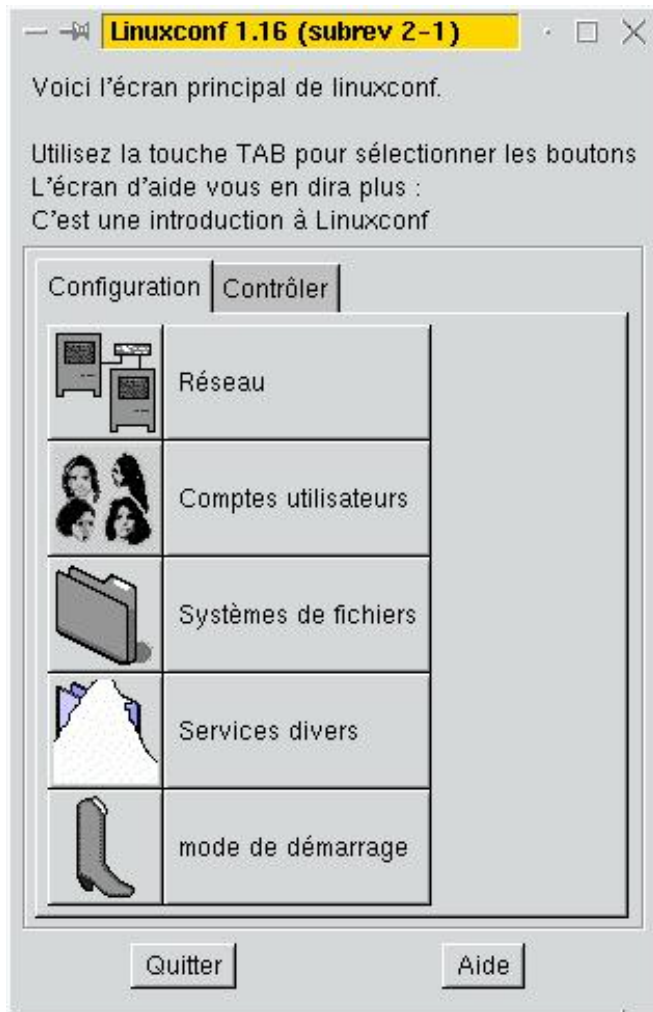
Démarrage par défaut : mode texte et réseau

LINUXCONF sous X-KDE

Dans une fenêtre de terminal `konsole` saisir **linuxconf** ou activer le lanceur s'il est présent sur le bureau.

La fenêtre d'accueil est composée d'onglets.

Pour l'essentiel, on retrouve les rubriques de la version texte.



TP

Lancer linuxconf en mode texte de préférence

- Examiner les comptes utilisateurs déjà créés. En créer de nouveaux (stagex / stgx).
- Créer un groupe stagiaire qui puisse accueillir tous les stagex
- Examiner le paramétrage du réseau, s'il est déjà effectué.
- Puis selon la situation, paramétrer la passerelle Internet ou la liaison PPP par modem.

Webmin, interface WEB d'administration

Installation

Téléchargement de l'archive webmin-0.78.tar.gz (~1,4 Mo) à <ftp://ftp.webmin.com/>
Puis :

```
# cp webmin-0.78.tar.gz /opt/webmin
   Copie du fichier dans un rép temporaire
```

```
# cd /opt/webmin
    décompression sur place --> création de webmin-0.78.tar
# gzip -d webmin-0.78.tar.gz
    désarchiver webmin-0.78.tar
# tar xvf webmin-0.78.tar
    installer cette application
# ./setup.sh
    --> installation de webmin dans /opt/webmin
```

Paramétrage

On valide les propositions d'installer les fichiers de configuration dans /etc/webmin
et les fichiers de log dans /var/webmin

Il faut indiquer le chemin de l'**interpréteur Perl**

--> /usr/bin/perl

Après vérification, il faut indiquer le nom de la distribution

--> 10 Mandrake Linux

Puis le numéro de la version

--> 3) pour la 6.1

Port du serveur Web accédé par Webmin

--> par défaut 10000

Nom de connexion à ce serveur

--> par défaut **admin**

Son mot de passe

--> admin

Nom de machine du serveur Web

--> par défaut, nom du serveur (ici p00)

Pas de support SSL présent

Démarrer le serveur webmin au boot

--> y

Désormais se connecter comme admin/admin à http://p00:10000

Essai

- Passer une requête d'URL http://p00:10000 (ou http://:10000) sur une station Windows du réseau.
- Le serveur exige le compte d'accès défini à l'installation.
Saisie du login et mot de passe : admin/admin
- Tester : suppression de qq comptes et rép perso.

Voici une partie de l'écran d'accueil de webmin

Webmin

Version 0.80 sur p00.maison.fr (Mandrake Linux 7.0)

[Site internet de](#)

[Webmin](#)

[Ecrire à l'auteur](#)

[Webmin](#)

Système

[Serveurs](#)

[Matériel](#)

[Autres](#)



[Actions de démarrage et d'arrêt](#)



[Composants Logiciels](#)



[Gestionnaire de processus](#)



[Montages disques et réseaux](#)



[Pages du manuel](#)



[Partage NFS](#)



[Quotas des Disques](#)



[SysV Init Configuration](#)



[System Logs](#)



[Tâches Cron](#)



[Utilisateurs et groupes](#)



Configuration réseau

Objectifs et remarques

Il s'agit de paramétrer un système LINUX pour qu'il soit connecté à un réseau local, et puisse éventuellement accéder au réseau Internet via un accès distant par routeur ou par modem.

Bien sûr un certain nombre d'éléments matériels (adaptateurs ...) et logiciels (démons lancés ..) doivent être installés, configurés et activés au démarrage de la machine.

Pour vérifier le bon fonctionnement du réseau local, un utilisateur peut :

- lancer des "ping" sur les machines voisines :

```
[stagex@pox]$ ping 10.194.2.10y
```

```
[stagex@pox]$ ping p0y si p0y figure dans le fichier /etc/hosts
```

- ouvrir une session telnet ou ftp sur un hôte dans lequel il possède un compte.(cf tp 1)
-



TP1

Vous êtes connectés sur p0x comme utilisateur stagexx.

Vous avez un compte ouvert sur la station p0y

Ouvrez sur **p0y** une session **ftp** et allez parcourir, selon l'étendue de vos droits, votre rép. perso sur p0y, et au delà.

Qq commandes ftp, en mode texte (bien sûr, pour un véritable travail, utiliser plutôt un [client graphique comme gftp](#), sous KDE)

```
[stagex@px] ftp p0y demande d'ouverture de connexion ftp sur p00
```

```
Name : (px:stagex) stagex
```

```
Password required for stagex : stgx
```

```
ftp>pwd
```

```
ftp>ls
```

```
ftp>cd /
```

```
.....
```

```
ftp>quit
```

Informations indispensables

- adresse IP de l'adaptateur réseau (une machine de type passerelle ayant 2 cartes)
Par exemple : 10.194.2.5 se composant de l'adresse du réseau (supposé de classe C) 192.168.1 et du numéro de la machine, ici 5 (de 1 à 254)

- Adresse de "boucle" : une machine isolée a toujours l'adresse 127.0.0.1, ce qui lui permet de se connecter à elle-même
- masque de sous-réseau (netmask) : ce qui détermine les adresses qu'il est possible d'attribuer aux machines de ce sous-réseau, ici 255.255.255.0
- adresse IP générale du sous-réseau : elle se déduit du masque et d'une adresse ; ici **10.194.2.0**
- nom complet : nom station + nom du domaine, ici **p0x.cfipen.fr**

Configuration initiale lors de l'installation

Prenons l'exemple d'une machine Linux, installé avec la distribution Mandrake (voir le chapitre [installation](#))

La configuration réseau comprend 2 parties :

- 1) Détection de l'adaptateur réseau Ethernet et intégration du pilote dans le noyau
Par exemple, au cfipen, pour une carte SMC Elite--> module wd.o !
- 2) Paramétrage réseau TCP/IP : il s'effectue à l'aide de 2 fenêtres de dialogue successives à renseigner.
Par exemple, au CFIPEN :
 - adresse ip : 10.194.2.100 à 10.194.2.119
 - masque de sous-réseau : 255.255.255.0
 - passerelle par défaut : 10.194.2.245 (adresse du routeur, ce pourrait être l'adresse d'un boîtier ISB 100)
 - serveur de nom primaire : 195.98.246.50 (Le DNS du fournisseur d'accès, ici ac-creteil.fr)
 - nom de domaine : cfipen.fr
 - nom de machine : p01.cfipen.fr à p19.cfipen.fr
 - 2ème serveur de nom :

Ces différents paramètres vont affecter divers fichiers de configuration que l'administrateur `root` doit connaître et savoir éventuellement modifier "à la main" ou par l'intermédiaire d'un [outil d'administration](#).

Interface Ethernet

Très important

Pour vérifier que l'adaptateur réseau est bien lié à la couche réseau du noyau Linux et activé, passer la commande **ifconfig**.

- **ifconfig nom-interface** renseigne sur l'interface, son paramétrage et son activité
`ifconfig lo` (interface loopback)
`ifconfig eth0` (interface Ethernet) donne la configuration irq, adresse E/S
- **ifconfig nom-interface adresse-IP** assigne cette adresse à l'interface et l'active

Si l'interface réseau n'est pas active, la commande **ifconfig eth0** renvoie par exemple :
ne.c : no PCI cards found. Use io=0xNNN values for ISA cards
eth0 : Device not found

Dans ce cas le plus simple est de rajouter l'indication manquante `io=0x300`, par exemple directement dans le fichier **/etc/conf.modules**, qui liste les modules chargées par le noyau, en particulier dans le cas d'une carte ISA, il doit contenir des lignes du genre :

```
alias eth0 wd                                assure la liaison de la 1ère interface réseau au pilote wd.o
                                              (pour pseudo carte Western Digital, alias SMC 8013)
options wd io=0x240 irq=5                    le paramètre io seul est exigé par le pilote
```

Sans rebooter ;~), il suffit de lancer les scripts d'initialisation réseau, (`/etc/rc.d/rc3.d/S50inet restart`) et la carte est enfin reconnue et liée au noyau !

Fichiers de configuration

Ils peuvent être examinés et parcourus en ligne de commande par la commande **less**.

Ou mieux à l'aide de l'utilitaire *Midnight Commander*.

Pour le lancer, passer la commande : **mc**

- mode affichage F3 : pour lire sans risquer de modifier (par inadvertance)
- mode édition F4 : pour modifier, F2 pour sauvegarder.

/etc/HOSTNAME *nom de la machine dans le domaine (obtenu aussi par la commande \$hostname)*
 pxx.cfipen.fr

/etc/hosts *table de correspondance des adresses IP des machines du sous-réseau et de leur nom d'hôtes.*

```
127.0.0.1          localhost
10.194.2.101       p01.cfipen.fr          p01
10.194.2.102       p02.cfipen.fr          p02
10.194.2.103       p03.cfipen.fr          p03
.....
```

Remarques :

Si on n'utilise pas de serveur de nom DNS local pour connaître la correspondance entre l'adresse **IP** de la machine et son **NOM**, il faut lister ainsi toutes les machines du sous-réseau et copier ce fichier **/etc/hosts** sur toutes les machines du sous-réseau.

Pour des réseaux de taille modeste, il est habituellement recommandé de maintenir cette liste des machines dans **/etc/hosts**, plutôt que de configurer et de gérer un serveur de noms.

Dès lors, on peut "pinguer" les stations du sous-réseau en utilisant les alias :

```
[jean@p01] ping p03
PING p03.cfipen.fr (10.194.2.103) : 56 data bytes
64 bytes from 10.194.2.103: ....
<Ctrl-C>
```

Dans **/etc/host.conf**, la ligne **order hosts, dns** indique que chaque machine cherche

d'abord l'adresse de la requête dans ce fichier.

L'adresse 127.0.0.1 est par convention l'adresse IP de la machine d'une machine isolée se connectant à elle-même (boucle interne); pour constater cette auto-connection : **ping localhost**

/etc/sysconfig/network-scripts/ifcfg-eth0 *fichier d'activation au démarrage de l'interface eth0*

```
DEVICE = eth0
IPADDR = 10.194.2.1xx
NETMASK = 255.255.255.0
NETWORK = 10.194.2.0
BROADCAST = 10.194.2.255
ONBOOT = yes
```

Plus généralement, les interfaces à activer automatiquement au boot sont définies dans les fichiers du répertoire

/etc/sysconfig/network-scripts/

/etc/sysconfig/network

```
NETWORKING = yes
FORWARD_IPV4 = no (empêche le transfert automatique des paquets ?)
HOSTNAME= pxx
DOMAINNAME = cfipen.fr
GATEWAY= 10.194.2.245
passerelle par défaut, par où chercher si l'adresse IP n'est pas dans le sous-réseau
GATEWAYDEV = eth0
```

/etc/host.conf *D'abord rechercher les hôtes dans le fichier /etc/hosts, puis dans le serveur de nom*
order hosts, bind # une machine peut avoir plusieurs adresses IP (dans le cas de multiples interfaces réseaux)
multi on

/etc/resolv.conf

nom de domaine local de l'ordinateur

domain cfipen.fr

adresse du serveur primaire DNS de Créteil

nameserver 195.98.246.50

liste de domaines à essayer, si le nom d'hôte ne précise pas son domaine

domainsearch ac-creteil.fr On peut mettre jusqu'à 3 serveurs de noms

/etc/networks

décrit les noms et adresses IP des différents sous-réseaux routables à partir de l'hôte

/etc/conf.modules

modules installés

```
alias eth0 wd  
options wd io=0x240
```



1. Examiner la configuration matérielle de l'interface réseau
 - `dmesg | less`
 - `ifconfig`
 2. Examen de la configuration logicielle
 - Editer les divers fichiers de **/etc** décrits précédemment
 - Voir avec l'utilitaire [Linuxconf](#), les diverses commandes de réglage.
Pour chacune, chercher la correspondance avec le fichier de configuration concerné.
-

Annexe

DNS =service de noms de domaine, est un logiciel contenant des tables de correspondance entre des adresses IP de machines et un ou plusieurs domaines.

A chaque requête Web notamment adressée au fournisseur d'accès académique, le navigateur client s'adresse à la machine abritant le DNS de Créteil, ie 195.98.246.50

Celle-ci se charge (éventuellement en demandant à son tour de l'aide à d'autres DNS) de trouver l'équivalent IP du nom de serveur présent dans l'URL de la requête.

Exemple si on est abonné à wanadoo, les requêtes s'adressent au nameserver (DNS) 193.252.19.3, paramètre mis dans les propriétés TCP/IP du paramétrage réseau sous Windows 9x, et pour nous, dans le fichier `/etc/resolv.conf`

Même si on dispose d'un serveur de noms local, il est conseillé de garder le fichier `/etc/hosts`.



Démarrer (sous) LINUX

et y rester, sans se faire jeter ...

Démarrage du noyau

- Après le chargement du bios, il y a exécution du chargeur de système [LILO](#) (LINUX LOader, installé le plus souvent dans le MBR). Le prompt **LILO :** s'affiche, et au bout de quelques secondes (durée réglable), le système par défaut s'installe, pour nous c'est *Linux, version Mandrake 6.1* .
- Usuellement, cette initialisation s'effectue en *niveau 3*, mode multi-utilisateurs avec les services réseau activés. Mais le niveau de chargement par défaut est réglable dans le fichier [inittab](#).
- Pendant ce court laps de temps, l'utilisateur peut donner des directives à LILO, notamment pour démarrer le système suivant un autre niveau que le niveau par défaut. Ainsi, on peut entrer :
 - `linux 1` (single), si on veut travailler exclusivement en mode mono-utilisateur
 - `linux 5`, pour démarrer le serveur X, et un Window Manager, directement **en niveau 5 graphique**
- Le compte-rendu de l'initialisation, en particulier des chargements des pilotes de périphériques, se trouve dans **/var/log/dmesg**
Les messages générés pendant l'initialisation du système sont consultables avec la commande **dmesg**

Connexion

Travailler sous le système LINUX, *même en dehors de tout contexte réseau*, **implique une connexion au système**. Une session monoposte n'est jamais anonyme.

Le processus d'identification est classique :

- Donner le nom d'utilisateur (login :)
- puis le mot de passe (password :)
- Si le compte est authentifié sur la machine, il y a rappel de la précédente connexion sous le même nom.
- Observer le prompt `[user@machine rép-perso]`
- celui-ci est modifiable; sa notation symbolique, `[\u@\h \W]\$`, est donnée par **echo \$PS1** voir sa définition dans **/etc/profile**

Lorsque nous sommes authentifiés, nous disposons des ressources du système selon les **permissions** (les droits des fichiers) que l'administrateur (le "**root**") a accordées.

Le shell

- En connexion, le système nous connaît, a ouvert une session de travail à notre nom, et attend nos directives , nos **commandes**.
- Plus précisément, il nous met sous le contrôle d'un programme qui joue le rôle d'*interpréteur de commandes* (semblable au rôle joué par *command.com* , vous vous souvenez ? ;-)
- Cette interface utilisateur est un programme qui s'appelle le **shell** (ce qu'il faut comprendre comme la "coquille qui enveloppe le noyau").
C'est notre interlocuteur, qui attend la saisie d'une ligne de commande et sa validation, pour analyser sa syntaxe et ... s'efforcer de comprendre notre demande pour l'exécuter (si possible !).
- Le shell lancé à chaque connexion peut être choisi lors de la création de l'utilisateur (par exemple avec Linuxconf, choisir dans une liste les interpréteurs disponibles).
Par défaut, il s'agit ici du shell BASH, le plus utilisé, lancé par la commande **/bin/bash**
- Pour connaître le shell et ses commandes internes, consulter son manuel : **man bash** (3923 lignes !)

Les commandes

Les commandes les plus simples sont les plus utilisées : elles sont internes au noyau , comme **ls**, **cp** ...

D'autres peuvent être des **alias**, des pseudos d'autres commandes.

Par exemple **ll** est défini comme *alias* de **ls -l** , **x** de **startx**, **m** de **mc -c** ...

Ce shell regarde si la commande que l'utilisateur lui lance est **interne**. Sinon, s'il s'agit d'un **alias** d'une autre commande. Sinon, il recherche un programme sur le système de fichiers, portant le nom de la commande, en se servant des chemins listés dans la variable **\$PATH**.

S'il trouve un tel programme, il l'exécute, en lui passant les arguments spécifiés sur la ligne de commande.



Exercice : Que font ces quelques commandes ?

pwd
who
ll
cd
echo Bonjour
echo -n Bonjour
echo \$PATH
clear
date
startx

Multi-connexions

Le système Linux est multi-utilisateurs

Comment le voir si on ne dispose que d'une seule machine ?

On peut se connecter plusieurs fois sur une même machine sous des identités différentes.

Pour cela on peut ouvrir des terminaux ou *consoles virtuelles* avec **Alt-Fx**, x=1 à 6, puis passer de l'une à l'autre avec la même commande.



Exo : prise en main de la multi-connexion

Si aucun compte utilisateur n'a encore été créé, il faut (forcément) se connecter comme administrateur **root**.

Créer tout de suite les 2 [comptes utilisateurs](#) **stagex** et **totox** (x=1 ..9, selon le numéro de votre station).

Effectuez alors les multi-connexions suivantes :

- **root** (passwd=**cfipen**) sur le terminal tty1
- utilisateur (login = **stagex**, password=**stgx**) sur le terminal tty2.
- dangereux (login = **totox**, password=**zigx**) sur le terminal tty3.

Puis :

- Remarquez le prompt **#** qui distingue **root** d'un quelconque utilisateur, de prompt **\$**
- Comment passer de l'une à l'autre de vos identités ?
Passez la commande **W**. Quelle est sa fonction ?
- Sous l'identité **stagex**, avez vous la permission de fouiner partout ? essayez donc d'aller dans **/root** (commande **cd /root**), le rép. personnel de **root** !
- **totox**, l'utilisateur à risque, essaie de supprimer quelques fichiers vitaux, comme **/etc/passwd**, le fichier définissant les comptes utilisateurs ou **/etc/inittab**, le fichier principal d'initialisation du système; y arrivera t-il ?

```
[totox@p0x /totox]$ cd /etc --> pour aller dans le rép. /etc
[totox@p0x /etc]$ rm passwd --> totox veut supprimer ce fichier
"rm: détruire le fichier protégé en écriture 'passwd'?" --> y , il ose confirmer !
"Permission non accordée" ouf !
[totox@p0x /etc]$ ll passwd --> voilà l'explication !
```



Attention !

Les manipulations précédentes apparemment bien innocentes comportaient déjà des risques importants ! Si par mégarde, sous l'identité de **root**, vous aviez supprimé quelques fichiers ... Il faut donc toujours se connecter et travailler comme utilisateur, *même sur son propre système*.

Si une tâche requérant les privilèges de **root** survient, on lance la commande **su** , qui place l'utilisateur en position de **root**, sur la même console (moyennant la fourniture du mot de passe); on quitte la commande su le plus vite possible par **exit**.



Manipulation

```
[stagex@p0x stagex]$ tty (où suis-je ?)
[stagex@p0x stagex]$ pwd (dans quel répertoire courant ?)
[stagex@p0x stagex]$ who am i (qui suis-je ?)
[stagex@p0x stagex]$ who (qui sommes-nous ?)
[stagex@p0x stagex]$ su (ouverture session superviseur)
password: cfipen_
[root@p0x stagex]$ who am i --> conclusion ?
```

Déconnexion et arrêt (volontaire)

- Pour se déconnecter, entrer **exit** ou **logout**
Cela relance l'attente de **login**.
- Evidemment, il ne faut pas éteindre brutalement ou rebooter sauvagement ! Chaque processus actif doit recevoir du noyau du système la directive de s'arrêter proprement, les systèmes de fichiers doivent être démontés.
- En cas de coupure brutale, le système effectuera des réparations au prochain démarrage, à l'aide de l'utilitaire **fsck**, avant de procéder à l'initialisation du système.
- Si un user qcq peut se connecter au démarrage, bien entendu pour des raisons de sécurité, l'arrêt est une tâche d'administration.
Pour arrêter le système, l'administrateur **root** lance l'une des commandes suivantes :
 - arrêt immédiat
halt (= shutdown -h now)
 - arrêt différé
shutdown -h <nb mn> il s'écoule <nb min> minutes entre l'avertissement et l'arrêt.
 - reboot
shutdown -r [<nb mn> | now] ou **reboot** ou **ctrl-alt-del**
- On peut éteindre à l'invite du message : *The system is halted*

Annexe 1 : Le processus de connexion

Lors de la création de son compte, un utilisateur est associé à un type de **shell**

Pour s'en convaincre consulter le fichier **/etc/passwd** : le dernier champ contient le nom du fichier exécutable (le shell par défaut) **/bin/bash**

L'interpréteur de commande associé est ainsi lancé automatiquement dès la saisie du login utilisateur.

Il poursuit sa configuration en exécutant des scripts globaux à tous les utilisateurs et des scripts liés au compte et qui permettent une personnalisation.

Enfin, il affiche le prompt et se met en attente de la lecture d'une commande.

Jusqu'à la commande **exit**, pour quitter le shell (ce qui équivaut à se déconnecter (logout))

Les scripts de connexion

1. d'abord le script **/etc/profile** *communs à tous les users y compris root*
2. celui-ci cherche à exécuter tous les scripts **/etc/profile.d/*.sh** (parcourir **alias.sh** et **numlock.sh**)
3. puis il y a exécution de **\$HOME/.bash_profile** (la variable **\$HOME** contient le chemin vers le répertoire personnel). Il s'agit ainsi d'un fichier de démarrage personnel et paramétrable.
4. A son tour il exécute **\$HOME/.bashrc** dans lequel il est recommandé de placer toutes les fonctions ou alias personnels (car **.bashrc** est exécuté dans tout shell)
5. Enfin le précédent exécute **/etc/bashrc**, dans lequel on place les alias globaux et la définition du prompt **\$PS1**
6. Puis le prompt utilisateur s'affiche et le shell attend une commande ...

Personnalisation du shell

/etc/bashrc est le dernier script d'initialisation du shell bash. Il contient des alias redéfinissables ou à compléter par l'utilisateur **root**.

Il suffit donc d'éditer et de compléter le fichier par défaut; par exemple :

vi /etc/bashrc (ou bien sûr utiliser l'éditeur de Midnight Commander, lancer **mc**)

```
alias l=""ls --color=tty -F -b -T 0"
```

```
alias ll="l -l"
```

```
alias lp="ll | more"
```

```
alias la="ll -a"
```

```
alias x="startx"
```

```
alias m="mc -c"
```

:wq (pour écrire dans le fichier et quitter vi)

Puis se reloguer (exit) pour relancer l'interpréteur du shell.

Personnalisation du login utilisateur

Chaque utilisateur peut ajouter des commandes shell au fichier de profil personnel, **~/.bash_profile**

Par exemple, voici ce que j'ai mis à la fin de ce fichier :

```
clear
```

```
salut="Bonjour $USER !"
```

```
# $USER contient le nom de connexion
```

```
echo "Nous sommes le $(date)"
```

```
# $( .. ) permet d'obtenir le résultat de l'exécution de la commande incluse
```

Annexe 2 : le processus de démarrage

Voici dans ses grandes lignes les phases du démarrage.

Démarrage du noyau

- A la mise sous tension, il y a la phase d'initialisation du **BIOS** de la carte mère : celle-ci fait l'inventaire de ses "petits", les divers périphériques dont elle se trouve dotée (bus, ram, disques, cartes ...).
Puis il part à la recherche d'un système d'exploitation sur l'un des périphériques accessibles ...
- Habituellement (si on ne démarre pas sur une disquette), le BIOS charge en mémoire le **MBR** (*Master Boot Record*, 1er secteur de la 1ère piste du 1er disque dur, 512 octets).
Supposons que le chargeur de systèmes **LILLO** (*Linux Loader*) s'y trouve, une première partie de LILLO est chargée et exécutée (A noter que le chargeur **LILLO** peut se trouver sur une disquette ou sur la partition active du disque)
- Sa tâche consiste à charger en mémoire la 2ème partie de LILLO (environ 5 Ko). Lors de cette phase, il y a affichage des lettres **LI**. S'il y a arrêt, c'est que LILLO n'arrive pas à s'exécuter (pb de géométrie du disque) que cette 2ème partie est ce qui va permettre à l'utilisateur de choisir le système à lancer.
- Le noyau de ce système quel qu'il soit (Linux, Windows ..) est décompressé "à la volée" et est chargé en mémoire. Ceci est accompagné de l'affichage du message "*Uncompressing Linux ...done. Now booting the kernel ..*"
- Dès lors c'est le noyau qui prend les affaires en main et inspecte son environnement matériel !

Les niveaux de fonctionnement

Ces 6 niveaux sont décrits au début du fichier **/etc/inittab**

- 0 : provoque un arrêt (shutdown) de la machine
- 1 : pour rentrer en mode mono-utilisateur, réservé à root
- 2 : mode multi-utilisateurs, sans NFS
- 3 : mode multi-utilisateurs avec tous les services réseaux
- 5 : démarrage du serveur graphique X11 en plus
- 6 : redémarrage de la machine (la commande **reboot** lance le niveau 6).

La ligne qui suit définit le niveau de fonctionnement par défaut au démarrage (*Default runlevel*), ici le niveau 3

id:3:initdefault

Donc pour changer de niveau par défaut, il suffit tout simplement de changer ce numéro !

Le premier processus, init

Le noyau du système chargé et décompressé, s'exécute et s'initialise : réservation mémoire, prise en compte de la zone d'échange (swap), détection du matériel et chargement des pilotes des périphériques, montage du système de fichiers et enfin lance le 1er processus **/sbin/init**

Le paramétrage de ce processus fondamental est entièrement assuré par l'exécution de ce fichier script **/etc/inittab** dont voici la suite :

```
# niveau d'exécution 3 par défaut
id:3:initdefault
# il y a ensuite exécution des scripts rc.sysinit
--> initialisation du PATH pour les autres scripts, activation swap,
montage systèmes fichiers, gestion des quotas..
si::sysinit:/etc/rc.d/rc.sysinit
# exécution du script etc/rc.d/rc avec le niveau en paramètre
--> lancement des divers services du niveau choisi, ici 3,
etc/rc.d/rc 3
# ceci lance tous les liens symboliques du rép rc3.d
```

/etc/rc.d/rc3.d/S*

par exemple /etc/rc.d/rc3.d/S01kernel.d* fait référence
 au script /etc/rc.d/init.d/kernel.d
 # les scripts de /etc/rc.d/init.d/ sont appelés avec un paramètre
 start, stop, status, restart
 Par exemple si on a modifié la configuration du serveur Samba dans le
 fichier **smb.conf**, il faut relancer ce service par la commande
 /etc/rc.d/init.d/smb restart
 # le fichier inittab se termine par
 # exécute xdm qui lance X et le login graphique, si le "runlevel" est 5
 x:5:respawn:/etc/X11/prefdm -nodaemon

Le dernier script d'initialisation à être exécuté est **/etc/rc.d/rc.local**

On y écrit à la fin d'éventuelles commandes pour charger des modules ou lancer des services supplémentaires.

**Manipulations**

1. Examiner le fichier **/var/log/dmesg** qui contient le compte-rendu de l'initialisation.
 Bizarrement, la commande **dmesg** nous en dit un peu plus, notamment sur l'interface réseau.
2. Examiner le fichier **/etc/inittab** avec **mc**
 Que faudrait-il y changer pour démarrer immédiatement au niveau X ?
3. Si vous êtes intéressé, suivez le cheminement décrit ci-dessus.

Annexe 3 : Vie et mort des processus

Un processus est un programme en cours d'exécution.

Le noyau Linux lance, gère les processus et contrôle leur échanges avec les périphériques. Il tient à jour une table des processus en exécution

Le premier processus, ancêtre de tous les autres est **init**. Tous les processus successifs sont créés par un processus parent et appartiennent à un utilisateur. Chacun est identifié par un numéro, son **PID**

Il peut être important de connaître le PID d'un processus, ne serait-ce pour pouvoir le "tuer", s'il ne répond plus et bloque une console

Voici comment consulter la table des processus et si besoin agir !

ps liste des processus

ps aux : donne tous les processus, avec leur numéro PID

ps aux | less : pour contrôler le défilement

ps aux | grep X11 : pour n'afficher que les lignes concernant le processus cherché.

kill PID : met fin normalement à la tâche

kill -9 PID : action si nécessaire encore plus radicale !

Sous X-KDE, on peut utiliser TaskManager, qui montre l'arborescence des processus.

**Manipulations**

1. Comment vérifier que le processus **init** est bien le tout premier lancé par le noyau ?
2. Connexions root dans tty1 et stagex dans tty2
 Expliquer ce que signifie la commande suivante et noter les numéros PID
ps aux | grep login
 Que se produira t-il si on supprime un processus login dans tty2 ? dans tty1 ? Vérifier.
3. Connexion comme stagex dans tty1 et dans tty2
 Lancer mc dans tty1, afficher un fichier

Dans `tty2`, repérer le numéro PID du processus `mc`, pour ensuite le supprimer `kill PID`.
Vérifier le résultat.

4. Lancer le serveur X-KDE par `startx`, passer en mode console dans un autre terminal, y repérer le PID de `kfm`, et le tuer.

Mais qu'avez-vous donc fait ? pouvez vous lancer des programmes ?

Ensuite, débarrassez-vous de `kpanel`.

Bravo, admirez votre oeuvre, comment allez-vous pouvoir quitter proprement le serveur **X** maintenant ?

Essayez de redémarrer le serveur X --> erreur : "remove /tmp/.X0-lock and start again".

Il n'y a qu'une solution, se débarrasser du processus parent qui est `/etc/X11/X ..`

```
ps aux |grep X ---> root PID=2128 /etc/X11/X ...
```

```
kill 2128
```

Annexe 4 : Le chargeur d'OS LILO

Exemple commenté de fichier `/etc/lilo.conf`

lilo est installé dans le MBR du 1er disque

pour l'installer sur disquette, mettre boot=/dev/fd0

```
boot = /dev/hda
```

transmission à LILO du fichier binaire contenant la description des noyaux

```
map=/boot/map
```

fichier binaire utilisé comme secteur de démarrage

```
install=/boot/boot.b
```

indique le label de l'image à charger, sinon c'est le premier rencontré

```
default=linux
```

affiche un texte explicatif au démarrage

```
message=/boot/message
```

vga spécifie l'affichage en mode texte

normal (80x25), extended (132x44 ou 132x60) ou ask (choix au démarrage)

```
vga = normal
```

active le mode interactif

```
prompt
```

chargement automatique dans 5 secondes du premier système

```
timeout = 50
```

fichier contenant l'image du noyau Linux à charger

```
image=/boot/vmlinuz-2.2.13-7mdk
```

```
label = linux
```

partition où se trouve la racine / du système de fichiers

Pour une disquette root=/dev/fd0

Si rien n'est spécifié, le système utilise le résultat de la commande `rdev`.

```
root = /dev/hda2
```

```
read-only
```

Le multi-boot

ou comment faire migrer en douceur son système de Windows9x vers Linux

- La cohabitation est tout-à-fait possible et même fructueuse entre les 2 systèmes sur la même machine.

On les installe dans 2 partitions différentes.

Attention ! apparemment, Windows exige d'occuper la 1ère partition primaire (correspondant à `/dev/hda1` sous

Linux) qu'il faut créer avec l'utilitaire DOS `fdisk`

Ensuite on installe normalement Linux, qui lui respecte l'environnement ...

Si on installe d'abord Linux, laisser une partition `/dev/hda1` de taille suffisante pour Windows.

Ne pas paniquer après l'installation de Windows (si tout s'est bien passé ...), Linux n'est plus accessible car Windows a écrasé **lilo**

Il faut alors rebooter Linux sur une disquette, puis root réinstalle lilo, en passant la commande **lilo**

- Au démarrage le chargeur de système **LILLO** permet de choisir.

Lorsque le message **LILLO** : apparaît, on peut saisir le nom, linux ou dos dans l'exemple ci-dessous.

La touche `tab` provoque l'affichage des systèmes disponibles sur la machine et le choix.

Au bout de 5 s (si `timeout=50`), le système par défaut démarre .

- Exemple de lilo.conf multi-systèmes

```
boot = /dev/hda
map=/boot/map
install=/boot/boot.b
prompt
timeout = 50
image=/boot/vmlinuz-2.2.13-7mdk
    label = linux
    root = /dev/hda2
    read-only
# other = place de la partition de l'"autre" système
other=/dev/hda5
    label=dos
# nom de la partition contenant la table de partition
table=/dev/hda
```

- Changer de système par défaut

- On repère les 2 paragraphes qui décrivent les boots . Le premier dans l'exemple (linux) est lancé par défaut.
- Pour changer de système au démarrage, il suffit donc de permuter tout simplement ces 2 paragraphes.

- Attention

1. Avant toute modification majeure, vérifier le bon fonctionnement de la disquette de redémarrage !
2. Après toute modification, passer la commande `/sbin/lilo`. Cela réécrit **LILLO** dans le MBR pour qu'elle puisse être prise en compte (n'oublions pas que `lilo.conf` ne sera pas lisible au boot !).

- Démarrer par un boot disquette

- Formater une disquette avec le système **ext2**.
utiliser l'utilitaire graphique *K/Utilitaire/Formatage de disquette* ou
mkfs (make filesystem): `# mkfs.ext2 /dev/fd0`
- Puis copier le noyau sur la disquette `cp /boot/vmlinuz-... /mnt/floppy`



Examiner le fichier **/etc/lilo.conf** de votre machine. Quelles en sont les infos essentielles ?

Voir l'utilitaire `linuxconf/Configuration/mode de démarrage`, plus convivial pour apporter des modifications à `lilo.conf`, et qui active les changements ...

Ou encore l'utilitaire **KLILLO** sous KDE .



Gestion des utilisateurs et des groupes

Qui est utilisateur ?

Le système, dès son installation, avant même la première connexion au système a créé des users système. Un utilisateur n'est donc pas uniquement une personne physique, le système a besoin d'utilisateurs pour sa gestion interne, notamment comme propriétaire des divers processus.

La commande `ps aux | less` montre qu'avant toute connexion d'utilisateur humain (repérée par les lignes `login --user`), `root` a lancé `init`, et la plupart des services, `crond`, `inetd`, `lpd`, `smbd`, ... , avant de lancer les connexions utilisateurs dans les consoles, y compris éventuellement la sienne !

Les principales commandes

<code>useradd</code> , <code>usermod</code> , <code>userdel</code>	gestion des comptes utilisateur
<code>groupadd</code> , <code>groupmod</code> , <code>groupdel</code>	gestion des groupes
<code>pwck</code> , <code>grpck</code>	vérification des fichiers
<code>passwd</code>	changer le mot de passe d'un utilisateur
<code>chfn</code> , <code>id</code> , <code>groups</code> , <code>finger</code>	utilitaires divers

Gestion des comptes

● Créer un compte pour un nouvel utilisateur

Cela signifie lui permettre d'être connu du poste local, s'y loguer, avoir un accès complet sur son rép. personnel.

Mais aussi dans une configuration réseau, de pouvoir se connecter à son compte par telnet et ftp, et de pouvoir bénéficier de services réseau de partage distant (sous Linux par NFS et sous Windows 9x par SMB).

● Pour créer l'utilisateur **stagem**, `root` passe la commande :

`useradd stagem`

Ceci crée :

- le répertoire personnel **`/home/stagem`**, portant par défaut le nom du compte
- une nouvelle entrée dans les 2 fichiers fondamentaux `/etc/passwd` et `/etc/group`.
- Pour connaître [les options](#) de `useradd` (indispensable pour gérer les comptes à l'aide de scripts)

● Pour lui attribuer le mot de passe :

`passwd stagem`

saisir 2 fois **`stgx`**

● Supprimer le compte d'un utilisateur (non connecté), au hasard .. **totox**.

`userdel [-r] totox`

L'option **`-r`** supprime aussi le rép. personnel et les fichiers de l'utilisateur

La commande supprime toute trace de l'utilisateur dans le fichier de configuration : `/etc/passwd` y compris dans les groupes d'utilisateurs.

- Modifier le compte de l'utilisateur toto
usermod [options] totox

Les options sont [les mêmes](#) que `useradd`

usermod -G stagiaire,prof stagex ajoute stagex dans les 2 groupes stagiaire et profs (qui doivent exister)



Manipulations

Sous l'identité de `root`

1. Créer quelques utilisateurs `stagey` et `totox`
2. Effectuer des vérifications : possibilité immédiate de se loguer sous ces comptes, création de leur rép. personnel dans `/home`.
3. Essayer de créer un compte déjà existant.
4. Supprimer sans regret le compte de `totox`. Son rép. personnel a-t-il été supprimé ?

Remarques

- Attention : si `root` passe la commande `passwd` il s'apprête à redéfinir son propre mot de passe !
- Un utilisateur quelconque ne peut pas créer de compte, même s'il a le privilège de faire partie du groupe `root`. A tester !
- Par contre, il peut modifier lui-même son mot de passe.
- Voir les diverses options avec **`useradd -h`**
- Pour une gestion sous interface graphique, voir le chapitre sur le nouvel outil [linuxconf](#)
- Attention ! Le compte créé permet à l'utilisateur d'accéder au système de fichier Linux (avec des droits que nous verrons). Pour pouvoir se connecter au réseau SAMBA, à partir d'une station distante Windows9x, il faut créer un compte Samba avec l'utilitaire `smbpasswd` (voir le chapitre [serveur Samba](#)).
A noter que `linuxconf` semble créer automatiquement les comptes Linux et Samba conjointement (voir [plus loin](#)).

Les groupes

- Un groupe est, aussi pour Linux, un ensemble d'utilisateurs qui partagent les mêmes fichiers et répertoires. Nous verrons que les fichiers accordent des droits d'accès réglables à ces groupes.
- Chaque utilisateur doit faire partie au moins d'un groupe, son **groupe primaire**. Celui-ci est défini au moment de la création du compte, et *par défaut*, l'utilisateur appartient à un nouveau groupe créé, portant son nom.
- Ainsi, dans `/etc/passwd` chaque utilisateur possède un groupe par défaut, précisé par son identifiant `gid` dans ce fichier.
- L'appartenance au groupe primaire n'étant pas exclusive, **tout utilisateur peut faire partie de plusieurs**

autres groupes, appelés ses *groupes secondaires*.

Mais le rôle joué par le groupe primaire demeure prépondérant, comme nous le verrons dans le système des permissions des fichiers.

- Pour lister tous les groupes (primaire et secondaires) d'un utilisateur :

groups stagex

- Pour créer un nouveau groupe

groupadd stagiaire

- Supprimer un groupe, au hasard .. encore `totox`.

groupdel totox

Le groupe est supprimé du fichier `/etc/group`.

- Pour ajouter un utilisateur à un groupe

Le plus simple est d'éditer le fichier `/etc/group` et d'ajouter une liste d'utilisateurs (séparés par des virgules) sur la ligne du groupe (ou utiliser `Linuxconf`).



Manipulations

Il s'agit de créer un groupe nommé **stagiaire** dont les membres sont les comptes **stagex**.

On donnera ensuite à ce groupe des droits complets sur un répertoire partagé.

Comme `root` :

1. créer le groupe `stagiaire` :
groupadd stagiaire
 2. ajouter quelques comptes `stagex` dans ce groupe
 3. vérifier le résultat avec la commande `groups`
-

Visite des coulisses

- Tout ce qui concerne la gestion et l'authentification des utilisateurs est inscrit dans un seul fichier **`/etc/passwd`**
- La gestion des groupes est assurée par **`/etc/group`**
- Les mots de passe cryptés sont maintenant placés dans **`/etc/shadow`**, par sécurité lisible seulement par `root`.

Structure de `/etc/passwd`

Ce fichier comprend 7 champs, séparés par le symbole :

1. nom de connexion
2. ancienne place du mot de passe crypté
3. numéro d'utilisateur **uid**, sa valeur est le véritable identifiant pour le système Linux; l'uid de `root` est 0, le système attribut conventionnellement un uid à partir de 500 aux comptes créés.
4. numéro de groupe **gid**, dans lequel se trouve l'utilisateur par défaut; le `gid` de `root` est 0, des groupes

d'utilisateurs au delà de 500

5. nom complet, il peut être suivi d'une liste de renseignements personnels (cf chfn)
6. rép. personnel (c'est également le rép. de connexion)
7. shell, interpréteur de commandes (par défaut /bin/bash)

Structure de /etc/group

Ce fichier comprend 4 champs, séparés par le symbole :

1. nom du groupe
2. x pour remplacer un mot de passe non attribué maintenant
3. numéro de groupe, c-à-d l'identifiant **gid**
4. la liste des membres du groupe



Manipulations

Editer ces fichiers (utiliser l'utilitaire mc, sélectionner et éditer avec F3)

Examiner les lignes correspondant aux comptes créés : où se trouvent les rép. personnels ? quel est leur groupe par défaut ?

Outils de gestion des comptes



Linuxconf

Se connecter comme `root` et lancer `linuxconf` en ligne de commande (on pourrait utiliser `linuxconf` sous X-KDE)

Section Comptes utilisateurs

- Sélectionner un compte et examiner sa définition actuelle sous l'interface de Linuxconf
Comparer avec son entrée dans `/etc/passwd`
- Ajouter un nouveau compte (`stagey/stgy`) en donnant seulement les nom de login et nom complet
A la validation, on observe que `Linuxconf` exécute 2 commandes `useradd` et `chage`. Puis on est averti que le mot de passe a bien été enregistré dans `passwd` et `smbpasswd`
- Pour connaître le rôle de `chage`, consulter le manuel et le fichier `/etc/shadow` Créer maintenant un autre compte (`toto`) en précisant le groupe primaire (`zig`), et des groupes secondaires (`stagiaire`). Examiner la syntaxe des 2 lignes de commandes exécutées (`/usr/sbin/useradd` et `/usr/bin/chage`)
- Si le rep. de base n'est pas spécifié, par défaut création et attribution de `/home/stagex`
- On peut tout de suite placer l'utilisateur dans une liste de groupes (sans virgule)



Kuser

- Lancer sous X-KDE, la commande `K/Système/Gestionnaire d'utilisateurs`
- Utilisation à découvrir

Compléments

- La structure d'une ligne de `/etc/passwd` et de `/etc/group`

```
login:x:uid:gid:commentaires:home:shell
groupe:x:gid:liste-groupes-secondaires
```

- Options de la commande **useradd** (pour détails cf `man useradd`)

Nous avons jusqu'ici utilisé cette commande avec ses options par défaut.

La maîtrise de cette commande est indispensable pour écrire des scripts de génération automatique de comptes.

Syntaxe : `useradd [options] nom_login`

Exemple : `useradd toto -u 1200 -p moi -g 520 -G groupes -s /bin/bash`

Options :

<code>-u uid</code>	pour fixer l'identifiant uid
<code>-g groupe-primaire</code>	
<code>-G liste</code>	fixe l'appartenance de l'utilisateur à une liste de groupes secondaires (séparateur , sans espace)
<code>-s shell</code>	par défaut, attribution du shell par défaut bash
<code>-c commentaire</code>	
<code>-d rep. personnel</code>	par défaut dans le répertoire /home
<code>-e date-expiration</code>	fixe la date d'expiration du compte (format MM/JJ/AA)
<code>-m</code>	pour créer le répertoire personnel
<code>-k rep-skel</code>	recopie le contenu de rep-skel dans le rép. personnel, par défaut <code>/etc/skel</code>

- La recopie du répertoire `/etc/skel` est très important pour l'administrateur, car il lui permet de configurer de façon uniforme les sessions de travail des utilisateurs.
C'est ainsi que tous les utilisateurs qui passe en mode graphique KDE hérite du même bureau.



Manipulations

- Dans une console lancer l'utilitaire **mc** et parcourir le répertoire `/etc/skel`. Peut-on "deviner" la fonction de ces fichiers ?
- Dans une autre console, se connecter comme `stagex` puis lancer `X-KDE`.
- Effectuer quelques travaux, comme modifier le bureau, par exemple en ajoutant un lanceur, puis lancer une session WEB avec Nestcape, aller visiter quelques pages.
- Dans la première console, avec `mc`, comparer d'un côté le contenu du répertoire "modèle" `/etc/skel`, et de l'autre le contenu actuel de `/home/stagex`, en particulier :
 - les fichiers de configuration des lanceurs **.kde1nk**
 - le contenu de **.netscape** : fichier `bookmarks.html`, le stockage dans le cache ...

- Ouvrir quelques fichiers **/home/stagex/Desktop/xxxx.kde1nk**, y repérer les noms des images et l'exécutable lancé par le clic.
- Pour examiner les valeurs par défaut appliquées par **useradd** :
commande `useradd -D` ou éditer `/etc/default/useradd`
`GROUP=100` *identifiant du groupe primaire*
`HOME=/home` *racine des rép. personnels*
`INACTIVE=-1` *(nb de jours avant destruction du compte*
`EXPIRE=` *nb de jours avant expiration du mot de passe*
`SHELL=/bin/bash` *shell de connexion attribué au compte*
`SKEL=/etc/skel` *fichiers recopiés par défaut dans chaque rép. personnel*
- La commande **passwd**
Elle est chargée du cryptage du mot de passe dans `/etc/shadow`
Syntaxe : **passwd [option] nom-login**
Options
 - **--stdin**, la commande abandonne son caractère interactif habituel et examine son entrée standard pour s'en servir comme mot de passe. Très utile dans un script : `echo mot | passwd --stdin` (attention tout caractère est significatif, y compris les " ")
 - **-d** , pour supprimer le mot de passe, l'utilisateur pourra se connecter sans !
 - **-l** , pour verrouiller le compte et empêcher sa connexion.
 - **-u** , pour déverrouiller.
- Connaitre l'uid et le gid de l'utilisateur courant
Commande **id**
`uid=501(stage1) gid=501(stage1) groups=501(stage1), 504(stagiaire)`
- Pour décrire un utilisateur : **chfn**
Cette commande permet d'indiquer dans le champ numéro 5 du fichier `/etc/passwd` différentes informations sur un utilisateur, son nom complet, son bureau, ses numeros de téléphone (séparées par des virgules).



Manipulation

```
[toto1@p01 ~]# chfn
Changing finger information for toto1
Password : zig1
Name [toto1] : Monsieur TOTO
Office [] : professeur au lycee Papillon
Office Phone [] : 0199999999
Home Phone [] :
```

- Cryptage des mots de passe
Pour des questions de sécurité, les mots de passe cryptés ne sont stockés dans `/etc/passwd` qui doit etre accessible en lecture par tous. La commande **/usr/sbin/pwconv** est chargée de transférer les mots de passes cryptés, dans `/etc/shadow`. Pour plus de détails , consulter man `pwconv`

Permissions d'accès aux fichiers



Tout fichier du système appartient à la fois à un utilisateur (son "propriétaire") et à un groupe. Ainsi, pour chaque fichier le monde de ses utilisateurs potentiels est scindé en 3 catégories, nommées :

1. **u**, l'utilisateur normal, son propriétaire, bien souvent son créateur, qui n'a pas pour autant tous les droits sur lui !
2. **g**, son groupe, ensemble d'utilisateurs ayant parfois des "permissions" particulières.
3. **o**, tous les (others) autres.

Attention, l'utilisateur propriétaire et le groupe propriétaire du fichier peuvent être indépendants :

- le groupe propriétaire n'est pas forcément le groupe primaire de l'utilisateur propriétaire,
- et même, le propriétaire n'est pas forcément membre du groupe !

Mais (heureusement) une règle générale simple s'applique à la création de tout nouveau fichier (ou rép)

- son propriétaire est l'utilisateur (humain ou système) qui l'a créé
- son groupe est le groupe *primaire* de ce même utilisateur

Droits d'accès des utilisateurs aux fichiers

Généralités

Linux permet de spécifier les droits d'action sur un fichier, que peuvent exercer les utilisateurs des 3 catégories précédentes, ou plutôt les **permissions** que leurs accordent les fichiers et les répertoires.

Linux a repris les 3 protections d'UNIX sur les fichiers et les répertoires. Leur notation symbolique est :

1. **r**, lecture
2. **w**, écriture
3. **x**, exécution

De façon générale, ces permissions sont consultables complètement par la commande : **ls -l**

Rappel : **ll** est un alias plus court, pour la commande **ls -l**

Par exemple :

```
[stagex@p0x stagex] ll *.html
-rw-r--r-- 1 stagex stagex 1200 oct 19 12 : 39 amoi.html
```

Description globale

On trouve de gauche à droite

- le 1er caractère indique la nature du fichier
"-" fichier normal, "**d**" un fichier répertoire, "**l**" un lien.
- le système de droits est spécifié symboliquement par les 9 attributs suivants, correspondants aux 3 catégories d'utilisateurs du fichier.

```

. . . | . . . | . . .
u   g   o

```

La section **U** fixe les droits accordés au propriétaire du fichier.

La section **G** fixe les droits accordés aux utilisateurs faisant partie du groupe auquel appartient le fichier.

La section **O** fixe les droits des autres utilisateurs.

- nombre de liens sur le fichier
1 signifie que le fichier n'a aucun lien qui pointe vers lui, 2 (ou plus) signifiant qu'il existe un lien (ou plus) vers lui.
- le nom du propriétaire du fichier
- le nom du groupe propriétaire
- la date de dernière modification

- le nom complet du fichier

Permissions des fichiers normaux

Pour chaque fichier, les utilisateurs sont ainsi séparés en 3 catégories, le propriétaire, les membres du groupe et tous les autres. Les permissions accordées par le fichier à ces catégories sont complètement indépendantes mais leur signification est la même. Vis à vis de chacune de ces 3 catégories, on trouve dans l'ordre :

- le droit de lecture , afficher son contenu --> "r" si permis , "-" si refusé
- le droit d'écriture , modifier son contenu --> "w" si permis , "-" si refusé
- le droit d'exécution , pour un fichier script ou binaire --> "x" si permis , "-" si refusé

Exemples :

- Le fichier de démarrage `/etc/rc.d/rc.sysinit` possède les droits `rwX r-X r-X`
Tous les utilisateurs ont donc le droit de lire et d'exécuter ce fichier (ce qui est à éviter); seul root peut le modifier
- La table de montage `/etc/fstab` : `rw-r--r--` peut être lue par tous, modifiée uniquement par root

Afficher toutes les infos sur un fichier

La commande **stat** permet d'obtenir une information plus poussée sur un fichier.

Exemple : `stat /etc/passwd`

Permissions des répertoires

Pour les fichiers de type répertoire, la signification des attributs est différente de celle d'un fichier normal.

Mais elle est toujours identique pour les 3 catégories d'utilisateurs du répertoire.

La présence d'un tiret "-" signifie toujours l'absence complète de droits

- **r** : lire le contenu, la liste des fichiers (avec `ls` ou `dir`)
- **w** : modifier le contenu : droits de créer et de supprimer des fichiers dans le répertoire (avec `cp`, `mv`, `rm`)
- **x** : permet d'accéder aux fichiers du répertoire et de s'y déplacer (avec `cd`). Si on attribue **w**, il faut attribuer aussi **x** sur le répertoire.

Exemples :

Passer les commandes **cd** / puis **ls -l**, pour lister les répertoires situés à la racine.

- A qui appartiennent-ils ? Un user quelconque peut-il y créer des sous-rép. ?
- Commenter les 2 cas particuliers **/root** et **/tmp**



Attention !

on voit que le droit **w** est très étendu, et même dangereux quand il est accordé à un groupe, car un membre du groupe peut supprimer des fichiers dont il n'est pas propriétaire et sur lesquels il n'a même pas de droit d'écriture !

Remarque

Le droit **x** sur un répertoire est un préalable indispensable pour qu'un utilisateur (de la catégorie correspondante au positionnement du **x**), puisse exercer d'éventuels droits sur les fichiers contenus dans le répertoire.



Exercice 1

Changements des droits

*De façon générale, l'utilisateur qui crée un fichier en devient le propriétaire,
et le groupe auquel l'utilisateur appartient (au moment de la création) devient le groupe du fichier.*

Remarques préalables

- Mais les droits accordés au propriétaire, au groupe et aux autres dépendent du processus qui a créé le fichier et du masque des droits.

- D'autre part l'administrateur peut être amené à effectuer des changements de propriété (par exemple pour permettre un travail en groupe) et des changements de droits sur des ensembles de fichiers et de répertoires, les étendre ou les restreindre.
- Et root n'est pas soumis à ces restrictions, il a le pouvoir absolu sur ... le système de fichiers. En contre-partie il peut être considéré comme responsable de tout dysfonctionnement !

Changer le propriétaire ou le groupe propriétaire

- Changer le propriétaire
chown [-R] nv-user fichiers
Commande réservée au propriétaire actuel des fichiers ou des répertoires (et à root)
L'option **-R** (récursif) permet d'agir sur l'ensemble des sous-répertoires.
Exemple : **chown -R stage4 /home/stage1**
- Changer le groupe propriétaire
chgrp [-R] nv-groupe fichiers
Ceci doit être effectué par root ou le propriétaire, à condition que celui-ci soit membre du nouveau groupe.
Exemple : **chgrp -R stage4 /home/stage1**
- Changer les 2 en même temps
chown nv-user.nv-groupe fichiers
chown new-user.fichiers
Dans ce cas, en plus, le groupe propriétaire des fichiers est changé pour le groupe primaire du nouveau propriétaire.

Changer les permissions sur les fichiers

- Les droits d'accès peuvent être modifiés par le propriétaire des fichiers ou par root (ou équivalent, d'uid 0).
- La commande **chmod** (*change mode*, change le "mode" des fichiers) peut s'écrire de plusieurs façons équivalentes, sur le modèle :
chmod droits fichiers
Le paramètre *droits* permet de calculer les nouveaux droits d'accès.
- Ceux-ci peuvent s'obtenir de façon *relative*, par ajout (symbole +) ou retrait (-) par rapport aux droits existants, ou bien de façon *absolue*, en fixant les nouveaux droits qui remplacent les anciens (symbole =).

Ajout, retrait ou fixation des permissions

Pour chaque fichier, on désigne par :

- **u, g et o** les 3 catégories d'utilisateurs (user, group, other) et de plus par **a** (=all) tous les utilisateurs.
- **r, w, x** les 3 attributs de chaque fichier, pour chaque catégorie d'utilisateur.
- **+ - =** l'action d'ajouter, de retirer ou de fixer un droit, qui s'applique à chaque catégorie séparément.
- les changements, sur le modèle "*à quelle(s) catégorie(s), quelle action, quel(s) droit(s)*" sont alors notés symboliquement :
[u g o a] [+ - =] [r w x]
- par exemple **chmod u+x fichier** signifie "ajouter le droit d'exécution au propriétaire du fichier"
- on peut regrouper les catégories si on veut exercer la même action :
chmod ug+w fichier "ajouter le droit d'exécution au propriétaire et au groupe"
chmod go-rwx fichier "enlever tous droits d'accès à tous les utilisateurs, sauf au propriétaire"

Notation relative (aux droits existants)

- **chmod [-R] <action-droits> fichiers**
- L'option **-R** (récursif) permet de modifier les permissions de tous les sous-répertoires.
- exemple : **chmod [-R] go-rwx /home/toto** enlève tous les permissions d'accès des fichiers du rép. personnel de toto (et des sous-rép.), à tous sauf au propriétaire, c'est-à-dire **toto**.

Notation absolue

- Pour chaque groupe, elle permet de fixer les nouveaux droits qui remplacent les anciens. Si une catégorie n'est pas présente, ses anciens droits s'appliquent.
- **chmod u=rwx,g=rw,o=r fichiers** remplace les permissions précédentes des fichiers, en les fixant à **-rwxrw-r--**
Attention : aucun espace dans la liste des droits, pas même autour des éventuelles virgules
- **chmod u=rwx,g=r fichiers** fixe les permissions à **-rwxr--???** en ne changeant pas les permissions précédentes du

groupe *other*

- **chmod u=rwx,g=r,o= fichiers** fixe les permissions à -rwxr-----

Remarque importante

Le "super-utilisateur" root n'est pas soumis aux restrictions des permissions. Une petite expérience :

1. Vérifier que `/etc/shadow` est inaccessible même en lecture aux utilisateurs
2. Vérifier que ses permissions sont ----- ou 400 en octal, seul le propriétaire root peut lire
3. Root supprime ce droit de lecture : `chmod u-r /etc/shadow`
Vérifier `/etc/shadow`
4. Root peut le lire, le copier et le modifier, ce n'est bien sûr pas recommandé, mais root peut tout se permettre (raison de plus pour ne jamais se connecter root, sans nécessité !)
5. Mais bonne nouvelle, root peut donc retrouver de fichiers appartenant à des utilisateurs ayant perdu leurs droits d'accès !

```
[stagex@p00 stagex]$ cp ./bashrc ./bashrc1
[stagex@p00 stagex]$ chmod ugo= ./bashrc1 aucune permission sur le fichier !
[stagex@p00 stagex]$ cat ./bashrc1 bien sûr il est totalement protégé en lecture
[root@p00 stagex]# cat ./bashrc1 mais pas pour root !
```



Exercice 2



Exercice 3

Compléments .. indispensables

Notation octale des permissions

Il existe une autre façon d'indiquer les permissions de chaque catégorie, plus simple en utilisant la numération octale

Voici la table de correspondance entre les 8 chiffres en numérotation octale (base 8) et les 8 valeurs de droits fichiers. Par convention la présence d'un droit est noté 1, l'absence 0.

Binaire	-----	Droit	-----	Octal
000	-----	(---)	-----	0
001	-----	(-x)	-----	1
010	-----	(-w-)	-----	2
011	-----	(-wx)	-----	3
100	-----	(r--)	-----	4
101	-----	(r-x)	-----	5
110	-----	(rw-)	-----	6
111	-----	(rwx)	-----	7

Synthèse : notation globale pour les 3 catégories

propriétaire			groupe			autre		
lecture	écriture	exécution	lecture	écriture	exécution	lecture	écriture	exécution
400	200	100	40	20	10	4	2	1

Pour obtenir les permissions exprimées en octal, il suffit d'ajouter en octal les nombres de la table de correspondance ci-dessus, pour lesquels les droits sont positionnés.

Exemples

```
chmod 700 /home/rep-a-moi droits par défaut pour un rép. personnel.
ls -l /home/rep-a-moi
--> drwx-----
```

Les 2 commandes suivantes sont équivalentes :

```
chmod 764 test
chmod u=rwx,g=rw,o=r test
ls -l test
-rwxrw-r--
```

Le masque de protection umask

- Rappelons les règles simples de propriété qui s'appliquent à la création d'un fichier ou d'un répertoire :
 - son propriétaire est l'utilisateur qui l'a créé
 - son groupe est le groupe *primaire* de ce même utilisateur
- Mais quelles sont les permissions attribuées *par défaut* à l'utilisateur propriétaire, au groupe propriétaire et à tous les autres ?
Les permissions maximales accordées par un fichier et un répertoire sont 666 (-rw-rw-rw-) et 777 (-rwxrwxrwx).
On peut restreindre ces permissions lors de sa création. C'est le rôle de la commande **umask** de fixer les permissions *masquées*, autrement dit **les droits non accordés aux fichiers et répertoires lors de leur création**.
- Exemple de calcul de permissions effectives, affectées lors de la création d'un répertoire, par un utilisateur dont le masque de protection est **027**

```
777 = 111 111 111 permissions maxi = rwx rwx rwx
- 027 = 000 010 111 masque de protection
= 750 = 111 101 000 permissions effectives = rwx r-x ---
```
- La commande umask
 - **umask** affiche le masque de l'utilisateur actif
Quelles sont les valeurs des masques par défaut de **root** et des autres utilisateurs ?
 - **umask -S** affiche les permissions correspondantes au masque, sous forme symbolique.
 - **umask *masque*** fixe les permissions ultérieures de création des fichiers de l'utilisateur actif, conformément à *masque*, en notation octale.
Attention ! le changement ne s'applique qu'à la présente session.
 - Pour la rendre permanente, on peut intervenir sur un fichier *profile* :
 - Dans le fichier profil général **/etc/profile**, on peut modifier la règle habituelle :
if [\$UID == 0] ; then umask 022 ; else umask 077 ; fi
 - Pour agir au niveau des utilisateurs, ajouter la ligne **umask *masque*** dans le fichier de profil personnel
\$HOME/.bash_profile

Les droits étendus

Le droit SUID

- Sa présence permet à un fichier exécutable de **s'exécuter sous l'identité et donc les droits de son propriétaire**, à la place des droits de l'utilisateur actuel qui l'exécute.
- Il s'agit d'un dispositif de sécurité essentiel qui autorise un utilisateur quelconque (par rapport à la commande) à bénéficier de droits plus étendus que les siens (souvent ceux de root), pour exécuter la commande agir sur d'autres fichiers indispensables, juste le temps et sous le contrôle de l'exécution de la commande, SANS qu'il soit nécessaire d'attribuer ces droits en permanence sur les fichiers.
- Ce droit est noté symboliquement **S** et se positionne à la place du x du propriétaire u (mais sans écraser le droit x)
Sa valeur octale est 4000



Exemple significatif

Examiner les droits du fichier exécutable **/usr/bin/passwd**, qui permet de (re)définir un mot de passe et le comparer à ceux du fichier **/etc/shadow** qui contient les mots de passe cryptés.

Observez :

```
ll /etc/shadow
-r----- root root shadow
ll -l /usr/bin/passwd
-r-Sr-xr-x root bin passwd
```

Comme le droit **x** est accordé à tous, chacun peut donc exécuter la commande **passwd**, mais personne ne possède pas lui-même le droit d'écriture dans le fichier **/etc/shadow** qui doit le stocker.

Le positionnement du SUID permet d'agir en tant que **root** lors de la demande d'accès au fichier et comme **root** a tous les droits, il est alors possible de mettre à jour ce fichier des mots de passe.



Manipulation

Comment connaître les commandes comme **passwd**, qui offre cette permission SUID ? Voici plusieurs façons

```
cd /usr/bin
# grep filtre les lignes produites par ls en utilisant
# l'expression rationnelle ^...s
ls -l | grep "^...s"
# pour afficher tous les fichiers possédant le SUID
cd /
ls -lR | grep "^...s"
# recherche parmi les fichiers ordinaires ceux qui ont au moins le droit s
find / -type f -perm +4000
```

Le droit SGID

- Pour un fichier exécutable, il fonctionne de la même façon que le **SUID**, mais transposé aux membres du groupe.



Exemple

- Examiner les droits symboliques de la commande d'impression **/usr/bin/lpr**
- Quelle est sa valeur octale ?
- Si une imprimante a été installée, un répertoire **lp** a été créé dans **/var/spool/lpd** . Or la commande **lpr** écrit dans ce répertoire. Comment un utilisateur quelconque peut-il alors y écrire le fichier d'impression ?
- Positionné sur un répertoire, ce droit modifie le groupe propriétaire d'un fichier créé dans ce répertoire. Un fichier créé dans un tel répertoire, verra son groupe propriétaire modifié : Ce ne sera plus le groupe primaire du propriétaire qui l'a créé (règle habituelle), mais à la place, le groupe propriétaire du répertoire lui-même. Autrement dit, ce droit **s** posé sur un répertoire, met en place un mécanisme d'héritage de groupe, de répertoire conteneur à fichiers contenus.
- Notation symbolique **s**, mis à la place du **x** du groupe, valeur octale 2000

Le "sticky bit"

- Ce droit spécial, traduit en "bit collant", a surtout un rôle important sur les répertoires. Il régleme le droit **w** sur le répertoire, en **interdisant à un utilisateur quelconque de supprimer un fichier dont il n'est pas le propriétaire**
- Ce droit noté symboliquement **t** occupe par convention la place du droit **x** sur la catégorie **other** de ce répertoire, mais bien entendu il ne supprime pas le droit d'accès **x** (s'il est accordé). Justement, si ce droit **x** n'est pas accordé à la catégorie **other**, à la place de **t** c'est la lettre **T** qui apparaîtra. Sa valeur octale associée vaut 1000.

- Pour positionner ce droit :

chmod +t rep

--> dt rep

--> dT rep

*si le répertoire a le droit x pour tous
sinon*



- Exemple

Le sticky bit est présent sur le rép. /tmp

Pour quelle raison ?

ls -l /

.....

drwxrwxrwt root root tmp/



- [Exercice 4](#) : sécuriser le partage d'un répertoire.
-



Systèmes de fichiers LINUX

**Sous LINUX, TOUT EST FICHIER,
organisé suivant une UNIQUE ARBORESCENCE
(dont la racine est nommée / et dont l'administrateur est root)**

Systèmes de fichiers

Un système de fichiers est une façon d'organiser et de stocker une arborescence sur un support (disque, disquette, cd ...). Chaque OS propriétaire a développé sa propre organisation.

On peut faire cohabiter plusieurs systèmes dans des partitions d'un même disque.

Linux possède son système appelé **ext2** mais peut en gérer d'autres. La liste en est donnée dans `/proc/filesystems`

L'utilisateur peut donc accéder sous Linux à d'autres systèmes de fichiers, comme DOS, Vfat,...provenant d'un périphérique ou importé par le réseau.

Comme pour l'utilisateur *tout est fichier*, tous les systèmes de fichiers quels que soient leur emplacement physique doivent être intégrés dans l'UNIQUE arborescence logique du système Linux.

Cette arborescence peut donc être construite (et évoluer) à partir de diverses partitions qui peuvent être situées sur plusieurs disques. Cela réalise une intégration et une abstraction plus poussée que dans le monde Windows où les partitions et lecteurs auxquels sont affectées les lettres A: C: D: ... demeurent des entités séparées. Naturellement la partition sur laquelle est situé le répertoire racine joue un rôle particulier.

Le processus de **montage**, avec sa commande **mount**, décrite plus loin, est le moyen de faire correspondre parties de l'arborescence et partitions physiques de disque. Il permet de plus d'affecter tout système extérieur (disquette, cdrom, dk zip, rép. réseau ...) à un répertoire créé pour cela dans l'arborescence.

Il suffira ensuite de se déplacer à ce répertoire, appelé **point de montage**, en fait un répertoire "*d'accrochage*", pour accéder à ses fichiers (bien sûr, conformément aux permissions que possède l'utilisateur)

Les différentes catégories de fichiers

- fichiers normaux
 - * texte : courrier, sources des programmes, scripts, configuration ...
 - * exécutables : programmes en code binaire
- fichiers répertoires
 - ce sont des fichiers conteneurs qui contiennent des références à d'autres fichiers.
 - véritable charpente de l'arborescence, ils permettent d'organiser les fichiers par catégories
- fichiers spéciaux

situés dans **/dev**, ce sont les points d'accès préparés par le système aux périphériques. Le montage va réaliser une correspondance de ces fichiers spéciaux vers leur répertoire "point de montage".
 par exemple, le fichier **/dev/hda** permet l'accès et le chargement du 1er disque IDE

- fichiers liens symboliques

Ce sont des fichiers qui ne contiennent qu'une référence (un pointeur) à un autre fichier.

Cela permet d'utiliser un même fichier sous plusieurs noms sans avoir à le dupliquer sur le disque.

Arborescence du système Linux

La racine est le sommet de la hiérarchie des répertoires. Il s'agit d'une arborescence logique, indépendante de l'implantation physique des divers sous-répertoires, qui peut s'étendre sur plusieurs partitions incluses sur un ou plusieurs disques, et même sur des disques réseaux.

Sa structure est standard, avec des extensions imposées par les distributions.

Toute modification est de la compétence exclusive de l'administrateur, à l'exception des répertoires personnels situés dans **/home**.

Il est recommandé de respecter cette architecture standard.

Liste des répertoires principaux et leur rôle :

/ le répertoire racine

- **/bin** les fichiers exécutables (en binaire) (initialisation du système + commandes "essentielles")
- **/boot** le noyau **vmlinux** et les fichiers de démarrage
- **/dev** répertoire de fichiers spéciaux, qui servent de canaux de communication avec les périphériques (disques, adaptateur réseau, cartes son etc...)
- **/etc** les fichiers de configuration du système et les principaux scripts de paramétrage
 - */etc/rc.d scripts de démarrage du système*
 - */etc/X11 scripts de configuration du serveur X*
 - */etc/sysconfig configuration des périphériques*
 - */etc/cron description des tâches périodiques à effectuer*
 - */etc/skel fichiers copiés dans le rép. personnel d'un nouvel utilisateur*
- **/home** la racine des répertoires personnels des utilisateurs
- **/lib** les bibliothèques et les modules du noyau
- **/mnt** la racine des points de montage des systèmes de fichiers périphériques ou extérieurs (cd, disquette, nfs ..).
- **/opt** lieu d'installation d'applications supplémentaires (comme starOffice, java ..)
- **/root** répertoire personnel du super-utilisateur root
- **/sbin** les fichiers exécutables pour l'administration du système
- **/tmp** stockage des fichiers temporaires
- **/usr** programmes accessibles à tout utilisateur; sa structure reproduit celle de la racine /
- **/var** données variables liées à la machine (fichiers d'impression, traces de connexions http, smb)

.. dans **/var/log**)

- **/proc** ce pseudo-répertoire contient une "image" du système (**/proc/kcore** est l'image de la RAM)

Parcourir et lister les répertoires

Voici les commandes indispensables (suivies bien sûr d'une validation) pour visiter l'arborescence.

ls commande générale d'accès aux infos des fichiers du rép. courant
essayer **ls**, puis successivement **ls -l**, **ls -a**

ls rep

cd chemin le chemin peut être absolu (indiqué à partir de la racine) ou relatif (à partir du rép. courant)

cd ..

cd raccourci vers le rép. personnel

file fichier renseigne sur la nature du contenu du fichier

pwd donne le nom complet du rép. courant

mkdir rep pour créer un sous-rep du rep courant

rmdir rep pour supprimer un sous-rep vide



Expérimentations

Faire de nombreux essais avec ces commandes.

Faire le lien entre la commande **file** et le 1er caractère affiché sur chaque ligne par **ls -l**

Un utilisateur *stagex* peut-il créer des rép. un peu partout ? essayer par exemple dans **/etc** ou dans **/usr**

Monter un système de fichiers

- Comme le système de fichiers Linux se concentre dans une seule arborescence de fichiers, l'accès et l'utilisation de systèmes extérieurs (disques, disquettes, **cd ..**) doit s'effectuer par **intégration** de ces systèmes de fichiers dans le système fondamental "*racine*".
Ce mécanisme d'intégration, souple et paramétrable, s'appelle le **montage**.
- Techniquement, l'opération de montage consiste à mettre en relation :
 - un fichier de périphérique situé dans **/dev** (qui permet la communication physique avec les données du périphérique)
 - avec un noeud d'insertion dans l'arborescence, appelé son **point de montage**
- Naturellement le montage fondamental est celui du répertoire racine. Celui-ci a dû être déclaré

(obligatoirement) **après** le partitionnement des disques et **avant** toute installation sur disque !

- Il est fondamental de bien comprendre ce concept : il conditionne tout accès à une ressource externe, en particulier à des ressources réseau à d'autres disques Linux (voir le processus d'[exportation NFS](#) chez le serveur, complémentaire du montage chez le client de la ressource)

Commandes de montage/démontage

- Il est toujours possible de monter "à la main" les systèmes de fichiers stockés sur les périphériques disques, cd ... avec la commande interactive `mount/umount`
- Syntaxe générale :
`mount -t <type> -o options /dev/rep-spécial /mnt/rep-montage`
Si cette description est présente dans le fichier `/etc/fstab`, la commande peut être simplifiée
`mount /dev/rep-spécial` ou **`mount /mnt/rep-montage`**
- Les types principaux
ext2 (type par défaut), **vfat**, FAT16 ou FAT32 de Win95 ou Win98, **nfs**, système de fichiers distant situé sur un [serveur NFS](#)
- Les options
les options par défaut sont:
rw (accès complet), **suid** (les éventuels permissions SUID et SGID des fichiers seront pris en compte), **dev** (permettre l'utilisation des fichiers de périphériques, **exec** (permettre l'exécution de fichiers binaires)
- Exemples
 - `mount` liste tous les systèmes de fichiers actuellement montés
 - `mount -a` monter tous les systèmes au démarrage, exécute `/etc/rc.d/rc.sysinit`
 - `mount /dev/cdrom` monte le système du cd-rom (si décrit dans `fstab`)
 - `umount /mnt/floppy` démonte le système de fichiers disquette
 - `mount -t vfat -o uid=5001,gid=5000,umask=022 /dev/hda1 /mnt/disk-c` monter la partition Windows occupant la 1ère partition `/dev/hda1` dans le rép. `/mnt/disk-c`, avec les options : l'utilisateur d'uid 5001, et le groupe de gid 500, seront propriétaires de tous les fichiers, la création d'un fichier s'effectuera avec le umask 022, c'est-à-dire les permissions 755 (`rwxr-xr-x`).

Gestion avec Linuxconf

Installer une nouvelle partition

Dans certains cas il peut s'avérer indispensable d'étendre le système de fichiers sur un nouveau disque dur, ou une partition récupérée ...

L'objectif consiste à assigner à une sous-arborescence du système de fichiers, cette nouvelle ressource périphérique, par le processus de montage.

Soit une nouvelle partition **`/dev/hda3`**, jusqu'ici "libre", à monter sur **`/home`**.

1. Avec `fdisk`, lui affecter un système 83 linux
 2. La formater
`mkfs -t ext2 -c -v /dev/hda3`
formate en blocs de 1024 en vérifiant les blocs(-c), puis écrit la table des inodes.
 3. effectuer une copie
`cp -r /home /root`
le déplacement de **/home**, dans **/root** par exemple. En effet **/home** est présent actuellement sur **hda1**, et il va être ensuite physiquement affecté sur **hda3**
Les rép. personnels sont actuellement dans **/root/home**
 4. monter la partition **hda3** en **/home**
`mount /dev/hda3 /home`
Expliquer les résultats des commandes :
`df`
`ll /home`
 5. récupérer le contenu de **/home**
`mv /root/home/* /home`
 6. pour automatiser le montage de **/dev/hda3** lors d'un redémarrage du système, ajouter dans la table de montage **/etc/fstab** la ligne :
`/dev/hda3 /home ext2 defaults 1 2`
-

Compléments

Le fichier **/etc/fstab**

Rappel :

Le processus **init** (exécuté au démarrage), après chargement du noyau, vérifie les systèmes de fichiers déclarés dans la table du fichier et effectue leur éventuel montage automatique.

Ce fichier **/etc/fstab** constitue une véritable "*table de montage*". Il fait l'inventaire des divers systèmes de fichiers que le noyau Linux est susceptible de gérer, précise la façon de les monter, s'ils doivent l'être au démarrage, etc ..

Structure de fstab

Sur chaque ligne on trouve la description du montage d'un système, avec 6 champs :

1. nom du fichier spécial (ou du système distant)
2. nom du point de montage, habituellement un sous-rep (éventuellement à créer) de **/mnt**
3. le type de fichiers : **ext2** (Linux), **msdos**, **vfat** (Win9x), **ntfs** (NT), **iso9660** (Cd-rom), **nfs**
4. liste d'options de montage, séparés par des virgules
Les options par défaut sont **rw,suid, dev, exec, auto, nouser**
 - **auto/noauto** , pour demander/empêcher un montage automatique au démarrage
 - **user/nouser**, pour autoriser/interdire un user qq (pas le "root") à effectuer le montage

5. paramètre pour **dump** (commande de sauvegarde)

Une valeur 0 signifie que le système de fichiers ne sera pas sauvegardé lors d'un **dump**

6. paramètre pour (commande de vérification des fichiers)

Il indique l'ordre dans lequel **fsck** devra vérifier les fichiers, 1 en priorité (c'est normalement la partition racine /, 2 sinon, et 0 pour ne pas demander de vérification.

exemple 1

```
/dev/hda1      /mnt/diskc      vfat      user, auto, rw
```

signifie :

/dev/hda1 est le descripteur de périphérique 1ère partition du 1er disque IDE

/mnt/diskc est le répertoire de montage

vfat est le type de système de fichiers (autres ext2, msdos, iso9660, nfs, swap)

exemple 2

```
/dev/hdb1      /mnt/disk_d      vfat      user, auto
```

au lancement du système, ou par la commande **mount -a**, le système de fichiers Windows 95, installé sur la 1ère partition du 2ème disque (unité D:\), sera monté automatiquement par tous les utilisateurs et accessible dans le répertoire /mnt/disk_d

remarques

Les lignes contenant l'option **noauto** ne sont pas montées lors du démarrage du système, mais sont utiles pour renseigner sur les paramètres à appliquer lorsqu'on effectuera le montage.

L'option **user** est nécessaire pour indiquer que n'importe quel utilisateur pourra monter et démonter le périphérique; sinon cette tâche est réservé au root.

Pour le [montage de système distant nfs](#), les options **rsize** et **wsiz**e sont optionnelles, mais permettent d'augmenter sensiblement les performances.



Expérimentations

1. Editer **/etc/fstab** et comprendre ses directives
2. Monter et démonter "à la main" le périphérique **/dev/cdrom**. Peut-on enlever le cd-rom avant de le démonter ?
3. Faire les mêmes manip sous X-KDE



Les inodes

Chaque système de fichiers tient à jour une table des descripteurs des fichiers qu'utilise le système d'exploitation pour accéder aux fichiers.

Cette table se compose pour chaque fichier, d'une entrée appelée **inode**, repérée par un index appelé le **numéro d'inode**

La liste des systèmes de fichiers gérés par Linux est visible sur **/proc/filesystems**

Il existe un outil de vérification et de réparation des systèmes : **fsck**, qui s'effectue sur un système ou

sous-systèmes, un rép. **obligatoirement démonté**
 par exemple, pour vérifier le rép. des users : **fsck /home**

Les fichiers spéciaux de `/dev` comporte l'attribut b (mode bloc) ou c (mode caractères)

Les liens (ln)

Les liens sont utiles pour faire apparaître un même fichier dans plusieurs répertoires, ou sous des noms différents. Ils évitent les duplications et assurent la cohérence des mises à jour

On distingue en fait deux sortes de liens :

1. les liens durs associent deux ou plusieurs fichiers à un même espace sur le disque, les deux fichiers restant indépendants. Par exemple : `ln linux.txt /home/stagex/linux-lien-dur.txt`
 Le fichier `linux-test-lien-dur` est créé dans le répertoire `/home/stagex`. On peut constater que ces 2 fichiers ont la même taille. Au niveau gestion ils sont indépendants, tout en partageant le même espace disque et donc le même inode. Toute modification de l'un, modifie l'autre ! Mais la suppression de l'un, casse le lien, mais ne supprime pas physiquement l'autre.
2. Les liens symboliques
`[root@p0x /home/httpd/html/LinuxCours] ln -s index.html /home/jean/accueil.html` La commande `ls -F` passée dans le répertoire `/home/jean` montre que le fichier `accueil.html` pointe sur `index.html` (ainsi, une requête sur `accueil.html`, va ouvrir `index.html`)
 Le lien symbolique fait référence à un fichier dans un répertoire. La suppression du fichier source entraînera un changement de comportement du fichier lien qui sera considéré comme "cassé" ("broken").



Observations

Passez les commandes `adduser` et `useradd` : elles semblent identiques, puisqu'on obtient le même message d'aide !

Pour comprendre :

1. cherchez où elles se trouvent : `whereis ...`
2. examiner ces 2 fichiers : `ls -l add*` et `ls -l user*`
3. comparez les tailles, les droits. Conclusion.



Installation du service NFS

*Le service **NFS** (Network File System),
permet le partage d'un système de fichiers sur un réseau Linux*

Généralités

- Il s'agit du protocole standard de partage réseau entre machines Unix, créé par SUN vers 1980. Il comprend l'ajout de fonctionnalités supplémentaires (dans la couche session au dessus de TCP/IP), les **RPC** (=Remote Procedure Calls)
- Donc une machine joue le rôle de serveur de fichiers. Elle est appelée **serveur NFS**, et
 - on dit qu'elle **exporte** tout (arborescence racine /) ou partie de son système de fichiers,
 - en le partageant par une liste de stations accessibles par réseau,
 - en installant toutefois des restrictions d'accès.
- Comme toute ressource extérieure doit être intégrée dans le système de fichiers Linux, cet accès ne pourra être permis qu'à l'aide d'un processus de montage : une partie de l'arborescence d'une machine Linux "serveur", est exportée ce qui lui permet d'être intégré dans le système de fichiers d'une machine Linux "cliente".
- L'utilisateur peut monter cette arborescence exportée par le serveur, sur un point de montage, de façon tout-à-fait semblable au [montage de systèmes de fichiers](#) des divers périphériques. Le montage peut s'effectuer en cours de session de travail par la commande interactive `mount`.
- Mais dans un cadre de travail stable, où le serveur est dédié, il est souhaitable de monter la ressource NFS au démarrage. Il suffit pour cela d'inclure la description du montage sur une ligne de `/etc/fstab`. On peut comparer le processus à la "connexion à un lecteur réseau" sur d'autres systèmes.
- Dès lors, pour l'utilisateur sur la machine cliente, la ressource est accessible comme si elle résidait sur un périphérique local.

Installation

Sur le serveur NFS

- Tout d'abord, les services **portmap** qui gère les connexions RPC, et **nfs** ont dû être installés initialement, sinon le faire (sur une Mandrake 6.1, il s'agit des packages `portmap-4.0 .. et knfsd-1.4.1-2mdk.i586.rpm`
Vérifier avec l'utilitaire **ntsysv** que les services `portmap` et `nfs` sont bien activés automatiquement au démarrage.
Pour vérifier que les "démons" correspondant sont en exécution : `ps aux | grep portmap`
- Sur le même modèle de fonctionnement que le serveur SMB, il faut mettre en activité ou relancer

les divers *daemons* qui mettent en service le protocole `rpc` (liste avec `whereis rpc`, fichier de configuration `/etc/rpc`) par

`/etc/rc.d/init.d/nfs restart`

- Le fichier **`/etc/exports`**.

Ce fichier (à créer s'il est absent) contient la liste des exportations.

Sur chaque ligne, on précise un répertoire du système de fichiers, suivi par la liste des machines distantes clientes autorisées à les monter.

Exemple de fichier **`/etc/exports`** sur `p01`

```
/home p02(ro) p03(ro)
/usr/bin p02(ro) p03(ro)
/home/httpd p03(rw)
/usr/public p03(rw)
```

Selon certaines documentations, il faudrait aussi autoriser la connexion en ajoutant une ligne dans le fichier `hosts.allow` du style : `ALL: 10.194.2`

Sur la station cliente

- On crée un rép. de montage qu'on peut situer dans **`/mnt`**, par exemple sur la machine `p03` :

```
[root@p03 /]# mkdir /mnt/nfs-p00
```

- Puis on monte sur le point de montage précédent, la ressource `/home/httpd/html` exportée par **`p00`**

```
[root@p03 /]# mount -t nfs p00:/home/httpd /mnt/nfs-p00
```

l'utilisateur sur `p03` pourra alors mettre à jour le site WEB distant sur `p00`

- Syntaxe générale

`mount -t nfs nom-machine:arborescence point-montage`

- Respects par nfs des droits

Bien sûr les permissions des fichiers importés s'appliquent vis à vis de l'utilisateur, notamment en ce qui concerne la directive **`(rw)`**. On ne pourra mettre à jour sur la station cliente, un fichier exporté que s'il possède la permission **`w`** vis-à-vis de l'utilisateur.

- Automatisation du montage

Pour cela, il suffit d'ajouter le contenu de la commande précédente dans une ligne du fichier [`/etc/fstab`](#)

```
p01:/home/httpd /mnt/import nfs auto, user
```

- Autres paramètres de montage

`insecure` autorise les accès non authentifiés

Selon certaines documentations, il faudrait l'indication que le noyau supporte le système de fichier `nfs`; pour cela, il faudrait dans `/proc/filesystems`, la présence d'une ligne **`nodev nfs`**.



Configuration avec Linuxconf

Soit à configurer l'export : du serveur p00 ---> stations clientes p02, p03

Pour chaque rép à exporter, on précise la liste des clients :

/home	p02, p03	r
/usr/bin	p02, p03	r
/home/httpd	p03	rw
/usr/public	p03	rw

Après validation, on peut vérifier que la table suivante a été écrite dans le fichier **/etc/exports**

/home	p02(ro), p03(ro)
/usr/bin	p02(ro), p03(ro)
/home/httpd	p03(rw)
/usr/public	p03(rw)



Commandes fondamentales

Introduction aux commandes

- Documentation générale
 - En tout premier lieu consulter les HOWTO dans `/usr/doc/HOWTO`
S'ils n'ont pas été installés, monter le CD, et installer le paquetage Mandrake/RPMS/howto-htmlrpm
 - Rechercher la doc sur les paquetages installés regroupée dans `/usr/doc`
 - Consulter le "Man" (= manuel) : en ligne de commande, `man commande`
- La syntaxe générale :
commande [options] paramètres
En général, les options sont précédées du symbole - et peuvent être groupées (ex : `rpm -ivh <nom-package>`)
Les paramètres précisent les fichiers concernés.
- Historique des commandes :
les touches flèches haut et bas permettent de parcourir les dernières commandes de l'utilisateur, stockées dans le rép. personnel dans le fichier `/home/stagex/.bash_history`
- Rôle des alias
 - Grâce aux alias des commandes, l'utilisateur peut créer des noms de commandes, construites bien sûr par combinaison des commandes standard, et même renommer les commandes de base (ne pas en abuser !)
 - Exemples d'alias, destinés à simplifier la vie, comme **ll** à la place de **ls -l**, ou à mieux accueillir les nouveaux venus à Linux (qui ont encore la nostalgie du DOS) comme **cd..** à la place de **cd ..**
 - Pour voir leur définition (Mandrake 6.x), éditer les scripts `/etc/profile.d/alias.sh` et `/etc/bashrc`
 - Pour en voir la liste, commande **alias**
 - Pour en ajouter en cours de session, par exemple : **alias x="startx"**
 - Voir le chapitre [shell-bash](#), pour apprendre à ajouter des alias permanents.

Importance des options

[stagex@p0x] `cd` *se placer dans son rép personnel*

Comparer les effets de :

```
ls
ls -l      liste avec les attributs des fichiers
ls -a      liste complète, y compris les fichiers cachés, qui commencent par un point
ls -la
ls -R      liste "récursive" des contenus des sous-rép.
ls --help  pour tout savoir !
```

Parcours et gestion des répertoires

Voir dans le chapitre sur les [systèmes de fichiers](#)

A noter que la commande **mkdir** permet de créer plusieurs niveaux de répertoires :

`mkdir -p docs/notes` crée docs s'il n'existe pas, et le sous-rép. notes

Créer et consulter des fichiers textes

- **touch** `fich1 fich2` crée les 2 fichiers vides
- On peut lire, créer ou modifier des fichiers de textes (par ex. des scripts, des fichiers de configuration) avec **vi**, l'éditeur apparemment rustique mais irremplaçable, qui reste le préféré de beaucoup d'administrateurs Unix.

- Pour seulement consulter un fichier texte, le plus simple consiste à utiliser les commandes **cat** (texte court) ou **less** (texte long)
Par exemple, examiner le fichier d'initialisation par **less /etc/inittab**
- En mode graphique, sous X-KDE (lancer **startx**), l'éditeur **kedit** est automatiquement appelé quand on ouvre un fichier texte dans l'explorateur **kfm**
- On peut saisir directement un fichier texte à la console avec **cat** et l'opérateur de redirection **>**.
Pour finir la saisie taper **Ctrl-d**.



Saisie directe à la console.

Il s'agit de créer quelques lignes de texte saisies et sauvegardées dans le fichier *essai.txt* du rép. personnel.

```
cd          pour aller dans son rép. personnel
touch essai.txt  pour créer ce fichier vide
cat essai.txt    pour vérifier
cat > essai.txt  cat créerait le fichier s'il n'existait pas !
Je suis heureux d'apprendre
à travailler avec LINUX
Ctrl-d          pour terminer et enregistrer
cat >> essai.txt pour ajouter du texte à la suite
enfant libre et gratuit d'Internet.
Ctrl-d
cat essai.txt    pour afficher
```

Connaitre les utilisateurs

La commande fondamentale est **id** qui donne (par défaut d'options)
l'uid (N° identifiant), le gid (N° de son groupe primaire), et la liste de tous ses groupes.

- exemple
[stage1@p01]id
uid=501(stage1) gid=501(stage1) groups=501(stage1), 504(stagiaire)
- **id toto** renseigne sur toto
- **id -u** donne l'uid
- **id -gn** donne le nom de login

Recherche de fichiers

- **whereis commande** pour rechercher les fichiers exécutables, les fichiers de configuration, les sources et les pages de manuel de *commande*.
La recherche s'effectue dans **/bin, /usr/bin, /etc ...**
Les options **-b, -m** limitent à la recherche des fichiers exécutables, des pages de manuel.
- **find rep -name expression** permet de rechercher les fichiers dans le rép (ou à défaut dans le rép. courant) avec une expression pour sélectionner.



Exemples

```
whereis w
---> w: /usr/bin/w /usr/man/man1/w.1
whereis ftp
whereis -b ftp
```

```
find -name smb*           recherche d'un fichier de configuration
find /usr -name pine      localiser une application
find / -name grasp*
```

Gestion des fichiers

Les principales commandes sont :

- **cp** (copy, copier fichiers et répertoires)

On distingue 2 usages :

- **cp [option] source destination**
copie d'un seul fichier, en précisant le chemin et le nom du fichier destination
- **cp [option] ens-fichiers-source répertoire**
copie l'ensemble des fichiers dans le rép. spécifié, en gardant les noms
- Principales options :
 - R** , recopie récursive, permet de copier toute une arborescence
 - i** avertit l'utilisateur de l'existence d'un fichier du même nom et lui demande s'il veut le remplacer.
 - p** effectue une copie en gardant le propriétaire et le groupe d'origine.
 - v** affiche en clair le nom des fichiers copiés.
- Exemples
`cp -R /home /root/tmp`, crée une copie dans `/root/tmp/home`
Dans la doc (man cp), on recommande d'utiliser `-R` et non `-r`

- **rm** (remove, supprimer des fichiers)

- **rm [option] fichiers**
- Attention, cette commande est très dangereuse. Les suppressions sont définitives, il n'y a de . Eviter de l'utiliser en tant que root !
- L'option **-i** a été ajoutée d'office dans un alias pour demander confirmation à l'utilisateur, pour chaque fichier.
- Options
 - r** permet de supprimer un répertoire et ses sous répertoires (attention TRÈS dangereux)
 - f** (force) permet de supprimer les fichiers protégés en écriture **sans** demande de confirmation. Cela permet d'inhiber l'option **-i** et de gagner du temps .. sauf dégats !
- Exemples
`rm -r /home/toto/tmp`, demande à l'utilisateur la permission de supprimer les fichiers un par un, et ne supprime pas le rep. s'il n'est pas vide.
`rm -rf /home/toto/tmp`, détruit sans préavis l'arborescence (si on en a le droit !)

- **mv** (move, renommer ou déplacer)

- **mv [option] source destination**
renomme simplement le fichier source, ce qui est un déplacement de nom ...
exemple : `mv toto titi`
- **mv [option] fichiers répertoire**
déplace les fichiers sources dans le répertoire, en gardant les noms.
exemple : `mv /home/jean/images/*.jpg /tmp/`
- Principales options :
 - b** (b=backup) effectue une sauvegarde des fichiers avant de les déplacer. La copie porte le même nom suivi d'un tilde.
 - i** (i=interactive) demande confirmation avant pour chaque fichier.
 - u** (u=update) pour ne pas supprimer le fichier si sa date de modification est postérieure à celle du fichier

remplaçant.

Gestion des processus

- La commande **ps** permet de connaître les processus actifs à un moment donné, associés à un terminal et lancés par l'utilisateur courant.
 - Pour obtenir tous les processus, y compris ceux lancés par le système et connaître l'utilisateur associé à chacun on ajoute l'option **aux**.
 - **kill num-pid**, pour faire terminer un programme, **kill -9 num-pid** pour le "tuer", s'il ne répond plus (le programme, pas le système ;-)
 - Description des colonnes de la commande **ps aux**.
 - "USER" à quel utilisateur appartient le processus.
 - "PID" est le numéro qui identifie le processus
 - "%CPU" en % les ressources du microprocesseur utilisées par le processus.
 - "%MEM" en % les ressources en mémoire vive utilisées par le processus.
 - "RSS" mémoire réellement utilisée en ko par le processus.
 - "START" l'heure à laquelle le processus a été lancé.
-

Commandes de filtre

sort	tri
more, less	pagination
grep	recherche d'une ligne dans un fichier texte
cut	effectue la projection d'un fichier selon une colonne
tr	effectue des remplacements de caractères
sed	effectue des modifications sur les ligne du fichier

Voir le chapitre [Introduction aux filtres](#)

Divers

- **write user [tty]** permet d'envoyer un message à **user** sur la console **tty** , bien sûr s'il est actuellement connecté, et si l'on est autorisé (pour cela : **mesg y**).
- **wall** envoie immédiatement un message sur les terminaux de tous les utilisateurs connectés (à condition que **mesg** soit positionné à **yes**; pour cela : **mesg y**)
- **sdiff** permet de comparer 2 fichiers et de détecter leurs différences.



TP extension du système de fichiers

Installer une nouvelle partition

I. Objectif

- On sait que les utilisateurs ont une vision logique du système de fichiers sous forme d'une seule arborescence, ce qui leur masque les particularités d'implémentation physique des fichiers sur disque(s).
Mais en réalité cette arborescence est en général le résultat d'un montage par emboîtement de plusieurs arborescences secondaires au système de fichiers racine
- Dans certains cas il peut s'avérer indispensable d'étendre le système de fichiers sur un nouveau disque dur, ou sur une partition installée dans une partie de disque inutilisée jusqu'alors ou récupérée ...
On rappelle qu'un système de fichiers ne peut pas s'étendre sur plusieurs partitions, a fortiori sur plusieurs disques physiques.
- Outre ces obligations, il y a des avantages à disposer Linux sur plusieurs partitions :
En cas (rare) d'endommagement d'un système de fichiers, le dégât est limité à la partition. L'administrateur peut mieux maîtriser le système et les espaces utilisateurs qu'il est souvent nécessaire d'augmenter de capacité
- Dans ce TP, il s'agit de décharger la partition "racine" éventuellement encombrée, des fichiers de l'un de ses répertoires principaux, afin de les stocker sur la nouvelle partition. Ensuite, il faudra récupérer les fichiers et procéder au nouveau montage automatique des partitions en modifiant **/etc/fstab**
- Pour connaître l'état du disque et en particulier pour savoir s'il reste de la place non affectée à un système, passer la commande `fdisk /dev/hda`, puis la commande **p** Supposons la situation initiale suivante :

```

/dev/hda1 200 Mo swap
/dev/hda2 2,5 Go linux partition montée en /
/dev/hda5 500 Mo linux partition montée en /home
le reste (au moins 1 Go) est libre

```
- Dans ce qui suit, nous prenons l'exemple du répertoire **/var** qui sera détaché de la partition racine, pour être physiquement stocké dans une nouvelle partition.

II. Sauvegarde des données à déplacer

- Comme les fichiers de **/var** actuellement situés dans la partition **/ racine** seront plus tard inaccessibles, il faut d'abord les copier ailleurs, dans une autre partition ou un autre support (lecteur zip par exemple) ou sur une partition réseau (NFS, Samba)
Conseil : il est préférable de copier par prudence (on effacera ensuite pour récupérer la place) et de pas déplacer comme cela serait plus adapté.

- On peut aussi utiliser `mc` dont la fonction copie (F5) préserve les permissions des fichiers.
- Dans notre exemple, effectuons une copie de **/var**, dans **/root/var**
Copie de toute l'arborescence (option **-R**) en préservant les permissions actuelles (option **-p**), pour plus de détails voir `cp --h | less`
cp -pR /var /root

III. Repartitionnement du disque

- S'il s'agit d'intervenir (supprimer, modifier, réaffecter) sur une partition existante, la démonter au préalable.
- Lancer **fdisk /dev/hda**, et créer une nouvelle partition :
device **/dev/hda6**, taille 300M, système linux native (code hexa 83)
Ecrire la nouvelle table de partition et quitter `fdisk`

IV. Intégration de la nouvelle partition

- Formater la nouvelle partition
mke2fs /dev/hda6
formate la partition et crée un système de fichiers `ext2`, puis écrit la table des inodes.
- Monter la nouvelle partition `hda6` en `/var`
mount /dev/hda6 /var
Pourquoi ce montage doit-il être effectué obligatoirement maintenant ?
Vérifier son accessibilité et expliquer les résultats de ces commandes :

```
du /var
df
ll /var
```

- Récupérer les données de `/var`
cp -pR /root/var/* /var
on copie (ici aussi par sécurité) le contenu de `/root/var` dans `/var`
- Automatiser ce montage de `/dev/hda6` lors d'un redémarrage du système, en ajoutant la directive suivante dans une ligne de la table de montage `/etc/fstab`:
`/dev/hda6 /var ext2 defaults 1 2`
- Par la suite, si tout est satisfaisant, supprimer les fichiers transitoires `rm /root/var/*` et quand le répertoire `/root/var` est vide, le supprimer : `rmdir /root/var`

V. Remarques importantes

- Si on supprime une partition auparavant montée, ne pas oublier d'intervenir dans `/etc/fstab` pour supprimer ou modifier la ligne correspondance.
Sinon, après un reboot, on a un message d'erreur sur l'impossibilité de monter et une invitation à effectuer une séance de maintenance par `root`.
- Attention, à veiller à garder l'intégrité de l'arborescence de la partition racine. Ainsi dans notre exemple, le répertoire **/var** doit exister dans la partition racine `/dev/hda2`, pour jouer le rôle de point de montage de la nouvelle arborescence.
Bien sûr ce répertoire `/var` est vide de fichiers, puisque ceux-ci seront affectés à la nouvelle partition `/dev/hda6`



Installation et archivage

Installation d'applications avec RPM

- L'utilitaire RPM (=RedHat Package Manager) gère une base de données des applications déjà installés. Il permet d'installer (et de désinstaller) facilement les nouvelles applications qui sont disponibles sous forme d'un fichier "paquetage". De plus, pendant une mise à jour RPM conserve les fichiers de configuration déjà présents.
- Syntaxe des paquetages
La syntaxe générale des paquetages à installer sur les machines à base de processeur Intel est `nom.version.i386.rpm`
Pour connaître toutes les options, voir **man rpm**
- Pour utiliser la commande **rpm** de façon aisée et transparente, il est recommandé de passer par l'utilitaire graphique **KPackage**
Mais le recours à la ligne de commande s'avère parfois indispensable, si on ne dispose pas de serveur X, ou si KPackage n'est pas installé !
- Principales options en ligne de commande
 - **rpm -q nomfichier.rpm**, pour obtenir de l'information
 - `rpm -q nomfichier.rpm` donne le numéro de version du programme s'il est installé, sinon renvoie le message *"package ... is not installed"*
 - `rpm -qa | less` donne la liste des programmes rpm installés
 - `rpm -qa | grep kernel` pour chercher les programmes du noyau
 - `rpm -ql kernel | less` donne la liste de tous les fichiers inclus dans les paquetages désignés, en particulier les modules installés dans `/lib/modules/...`
 - **rpm -i nomfichier.rpm**, commande générale d'installation
 - `rpm -i` cette commande, réservée à `root`, décompresse les programmes en les installant dans les bons répertoires.
 - les options `v` et `h`, facultatives, permettent de voir l'état d'avancement de l'installation.
 - `rpm -ivh -- nodeps nomfichier.rpm`, pour contourner le refus d'installer en raison de dépendances non satisfaites.
 - `rpm -ivh -- force nomfichier.rpm`, pour forcer l'installation en cas de conflit avec une version déjà installée
 - Exemple : pour installer les HOWTO en français, monter le cd-rom, aller dans `/Mandrake/RPMS/`, et passer la commande `rpm -ivh howto-french-*`
 - **rpm -U nomfichier.rpm**, commande de mise à jour d'un paquetage déjà installé.
L'ancienne version est d'abord retirée, tout en préservant les fichiers de configuration
 - **rpm -e nomfichier.rpm**, pour désinstaller (e=extract) un programme.

Cette commande controle les dépendances, et signale les autres programmes qui en ont besoin. Donc, attention à ne pas désinstaller des fichiers en dépendance.

- **rpm -V nomfichier.rpm**, cette commande compare les fichiers installés avec les fichiers d'origine du paquetage, pour vérifier que tous les fichiers d'un programme sont présents et pour connaître ceux qui ont été modifiés depuis

Sauvegarde et archivage avec tar

● Généralités

La commande **tar** (=Type ARchive) est une ancienne commande Unix qui permet aisément d'archiver, c'est-à-dire de réaliser la sauvegarde d'un ensemble de fichiers en un seul fichier, que l'on peut également compresser. Certaines applications et des mises à jour (les noyaux Linux notamment) ne sont livrées que sous forme soit binaire, soit de source à compiler, dans ce format (bien que les applications soient de plus en plus disponibles précompilées, prêtes à l'emploi, sous format [.rpm](#))

● Syntaxe

tar options fichiers

fichiers :

désigne un ensemble de fichiers ou toute une arborescence précédée d'un chemin absolu (à partir de /) ou relatif. Il est recommandé d'indiquer un chemin absolu qui sera conservé dans l'archive et permettra ensuite un désarchivage correctement positionné (sinon il y a installation conformément au chemin relatif conservé, ce qui nécessiterait un exact positionnement dans le système de fichiers).

options :

Les 3 premières -c -x -t spécifient les 3 types d'actions de la commande

- **-x** e**X**traire le contenu d'une archive
- **-c** Créer une nouvelle archive
- **-t** afficher seulement la liste du contenu de l'archive, sans l'extraire
- **-f fichier** indiquer le nom du **f**ichier archive
- **-v** mode ba**V**ard
- **-z** compresser ou décompresser en faisant appel à l'utilitaire [gzip](#)
- **-y** compresser ou décompresser avec l'utilitaire [bzip2](#)
- **--help** aide
- **-B** pour éviter le **b**locage en utilisant un pipe

● Utilisation et exemples

1. Création

`tar -cvf sauve.toto.tar /home/toto` effectue la sauvegarde de tous les fichiers du répertoire `/home/toto` dans le fichier `sauve.toto.tar` placé dans le rép. courant

`tar -cvf /tmp/sauve.toto.tar /home/toto` idem, mais le fichier archive est placé dans le rép. `/tmp`

`tar -c /home/toto > sauve.toto.tar` variante de la commande précédente

`tar -cvf sauve.toto.tar /home/toto`

`tar -cvzf sauve.toto.tar.gz /home/toto` effectue une compression en plus

2. Listage

`tar -tvf sauve.toto.tar` pour connaître l'arborescence regroupée dans le fichier archive, en particulier la place où sera installée son contenu lors du désarchivage.

L'utilitaire **mc**, avec sa fonction d'édition F3, permet d'effectuer le même listage de l'archive

3. Extraction

`tar -xvf sauve.toto.tar` exécute le désarchivage dans le répertoire courant.

si l'archive a été créée par `tar -cvf sauve.toto.tar /home/toto`, il faut se placer à la racine / pour restorer exactement le rép. perso de toto.

décompresse et désarchive

`tar -xvfz sauve.tar.gz home/toto/tmp` ne désarchive dans l'archive, que le rép. désigné

Compression avec gzip

● La commande gzip

Elle est utilisée pour compacter un fichier quelconque, et en particulier une archive `tar`.

Le décompactage se fait par la commande **gunzip**, ou de manière totalement équivalente par **gzip -d**.

Elle peut décompacter les fichiers `.gz`, mais aussi les fichiers `.z` , `.Z`

● Options

- **-1 . . . -9** fixe le niveau de compression
- **-d** décompresse
- **-c** écrit sur la sortie standard au lieu de remplacer le fichier d'origine (possibilité d'utiliser un tube)
- **-l** affiche des infos sur la dé/compression.
- **-r** dé/comprime tous les fichiers du rép. passé en argument.
- **-h** aide

● Exemples

- **gzip backup.tar /home/toto** compresse *backup.tar* et le remplace par le fichier *backup.tar.gz*, d'une taille beaucoup plus réduite.
Attention, le fichier d'origine est donc détruit !

- **gzip -9 *.txt** compresse au maximum chaque fichier **.txt** séparément, et les renomme en ajoutant le suffixe **.gz**

● Autre utilitaire

bzip2 admet la même syntaxe que **gzip**, mais compresse mieux avec un besoin accru de mémoire

TP 1

1. Sous l'identité **root**, créer le rep. **/home/archives/stagex**. Pourquoi nécessairement est-ce le travail de **root** ?
2. Sous quel masque **root** a t-il créé ce rép ? **stagex** va t-il pouvoir y archiver ses documents ?
3. Faire ensuite le nécessaire pour que l'user **stagex** puisse se réserver *exclusivement* l'accès et l'usage de son rép. personnel d'archivage
4. **stagex** archive dans **/home/archives/stagex**, sous le nom **sauve.stagex.tar**, son rep. personnel **/home/stagex**.
5. Puis il effectue maintenant des sauvegardes compressées, par **gzip** et **bzip2**, respectivement sous les noms **sauve.stagex.tar.gz** et **sauve.stagex.tar.bz2**
6. Vérifier l'existence et comparer les tailles des 3 archives obtenues.
En vérifier les contenus et le point de désarchivage par **tar -tvf ...** ou **mc**
7. **stagex _très maladroit_** détruit son rép. personnel **/home/stagex**
8. Heureusement, il peut effectuer un sauvetage ! comment ? aidez-le !

TP 2

1. Se connecter à la page d'accueil du serveur WEB (Apache), à l'URL **http://p00.fctice77.fr**
2. Sur une station Linux locale, télécharger le fichier **cours-linux.tgz** et le placer dans **/home/stagex/LinuxCours**, dans le répertoire personnel de l'utilisateur **stagex**
3. Décompresser, puis désarchiver le fichier. Relever les tailles successives, puis passer la commande **du ..** Consulter localement ce support de cours.
4. Mêmes questions sur une station Windows, à partir du fichier **cours-linux.zip**

Réponses aux questions

TP 1

1. Le rép. a les droits 755 pour **root**. Comme il n'est pas question que **stagex** fasse partie du même groupe de **root**, **stagex** est un autre user, sans droit d'écriture, c'est-à-dire de création de sous-rép.
2. Les rép. **/home/archives/stagex** doivent être créés par **root**, à charge ensuite que celui-ci accorde les permissions totales à chacun si les archives sont publiques, ou accorde le droit de

propriété de chaque stagex sur son rép. d'archivage /home/archives/stagex. On peut évidemment vérifier que sans cela

```
tar -cvf /home/archives/stage1/sauve.stage1.tar /home/stage1
```

provoque l'erreur *Permission non accordée*

3. Root change les propriétés et les droits : `chown stagex /home/archives/stagex`
`chgrp stagex /home/archives/stagex`
4. stagex se réserve tous ls droits exclusifs :
`chmod 700 /home/archives/stagex`
5. Le maladroit : `rm -r /home/stage1 *`
6. Sauvetage !

TP 2

1. Le répertoire courant contenant les fichiers à archiver ainsi que le sous-répertoire /images, le fichier `cours-linux.tgz` a été créé par la commande :
`tar czvf cours-linux.tgz *`, puis il a été déplacé dans /home/httpd/html/archives
2. Téléchargement sur la station Linux avec le client Netscape :
`http://p00` charge la page d'accueil du serveur
Dans le menu contextuel (clic droit maintenu) sur le lien, choisir *Enregistrer le lien sous* dans le répertoire /home/stagex/LinuxCours
3. Désarchiver avec la commande **`tar xzvf cours-linux.tgz`**



Impression sous Linux

Le système d'impression

Il est assuré par le "démon" **lpd** (=line printer daemon), lancé au démarrage.

Comme beaucoup de démons, il est géré par un script situé dans **/etc/rc.d/init.d/lpd**

Le script **lpd** attend un des arguments **start**, **stop** ou **status**, selon que l'on veut commander le démarrage du démon, son arrêt ou de l'info par le script

Il paraît que le démon est instable, il faut donc savoir le relancer

Lorsqu'il est rendu actif, ce démon **lpd** lit le fichier de configuration **/etc/printcap**, imprime les fichiers d'impression éventuellement en attente et se met à l'écoute de nouvelles tâches d'impression.

lpd active 2 autres démons *listen* et *accept*, qui sous-traitent les tâches d'impression.

Le sous-système d'impression sait gérer aussi bien des imprimantes connectés physiquement au serveur (imprimante locale), que des imprimantes accessibles par le réseau (imprimante lointaine). En particulier on peut envoyer des fichiers d'impression dans une file d'attente Linux, à partir de station Win95/98 via le protocole [Samba](#).

Les requêtes d'impression provenant du réseau sont écoutées sur le port 515 nommé **printer** (pour le voir éditer le fichier */etc/services*)



Manipulations

- arrêter le démon lpd
- demander le status
- redémarrer ce démon ..

Configuration du service d'impression

- La configuration lors de l'installation est recommandée.
- Le fichier de config **/etc/printcap** contient des informations sur les imprimantes rattachées localement ou lointaine avec lesquelles le démon doit communiquer.
- On conseille de ne pas directement le modifier (très risqué à cause de la difficulté de respecter les formats d'écriture).
- Le lire (mc -c) --> il contient la ligne **lp=/dev/lp0:** qui assigne à l'imprimante par défaut nommée **lp**, le fichier de périphérique **/dev/lp0**, **qui correspond au port parallèle LPT1**
Pour tous les détails voir le manuel : **man printcap**

Exemples

nom court de l'imprimante locale

lp:\

nom du répertoire de la file d'attente (sd=spool directory)

:sd=/var/spool/lpd/lp:\

la taille maximum du fichier est illimitée (car 0)

:mx#0:\

pas de page de séparation

:sh:\

nom du fichier spécial pour printer locale

:lp=/dev/lp0:\

nom du fichier de traitement du fichier

:if=/var/spool/lpd/lp/filter:

nom court de l'imprimante REMOTE

lp:

nom du répertoire de la file d'attente

:sd=/var/spool/lpd/lp:

.....

nom du serveur d'impression distant (rm=remote machine)

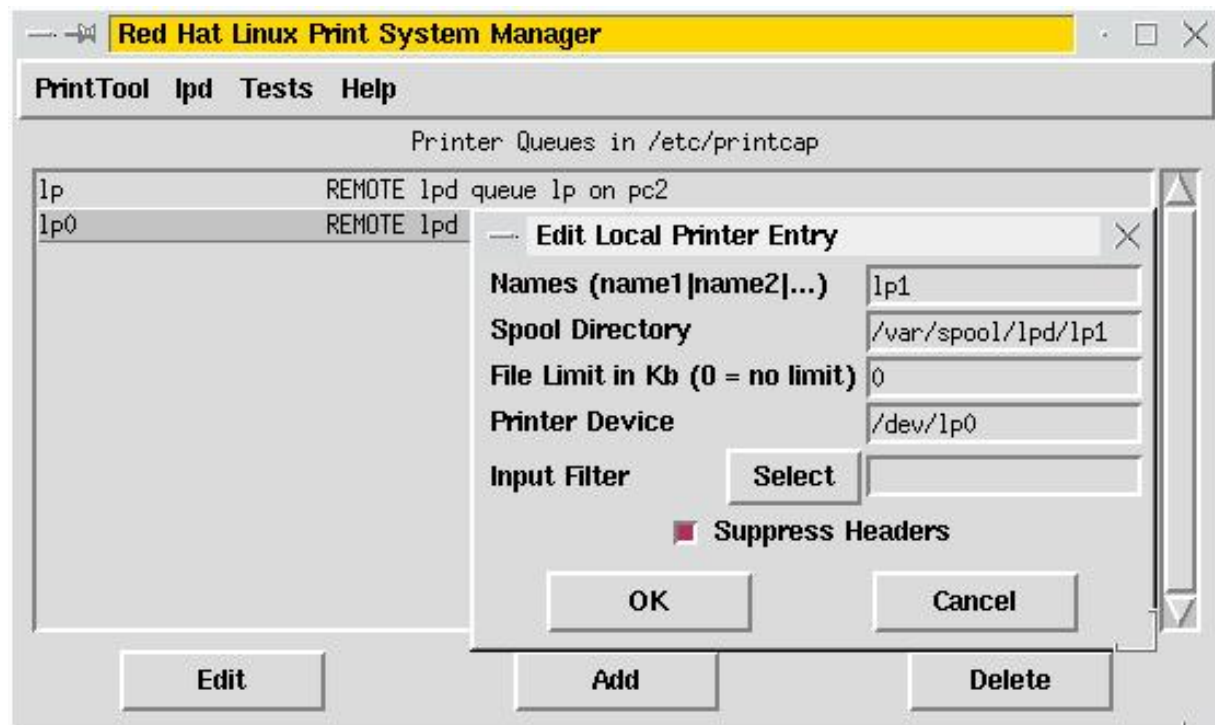
:rm=pc1.cfipen.fr:

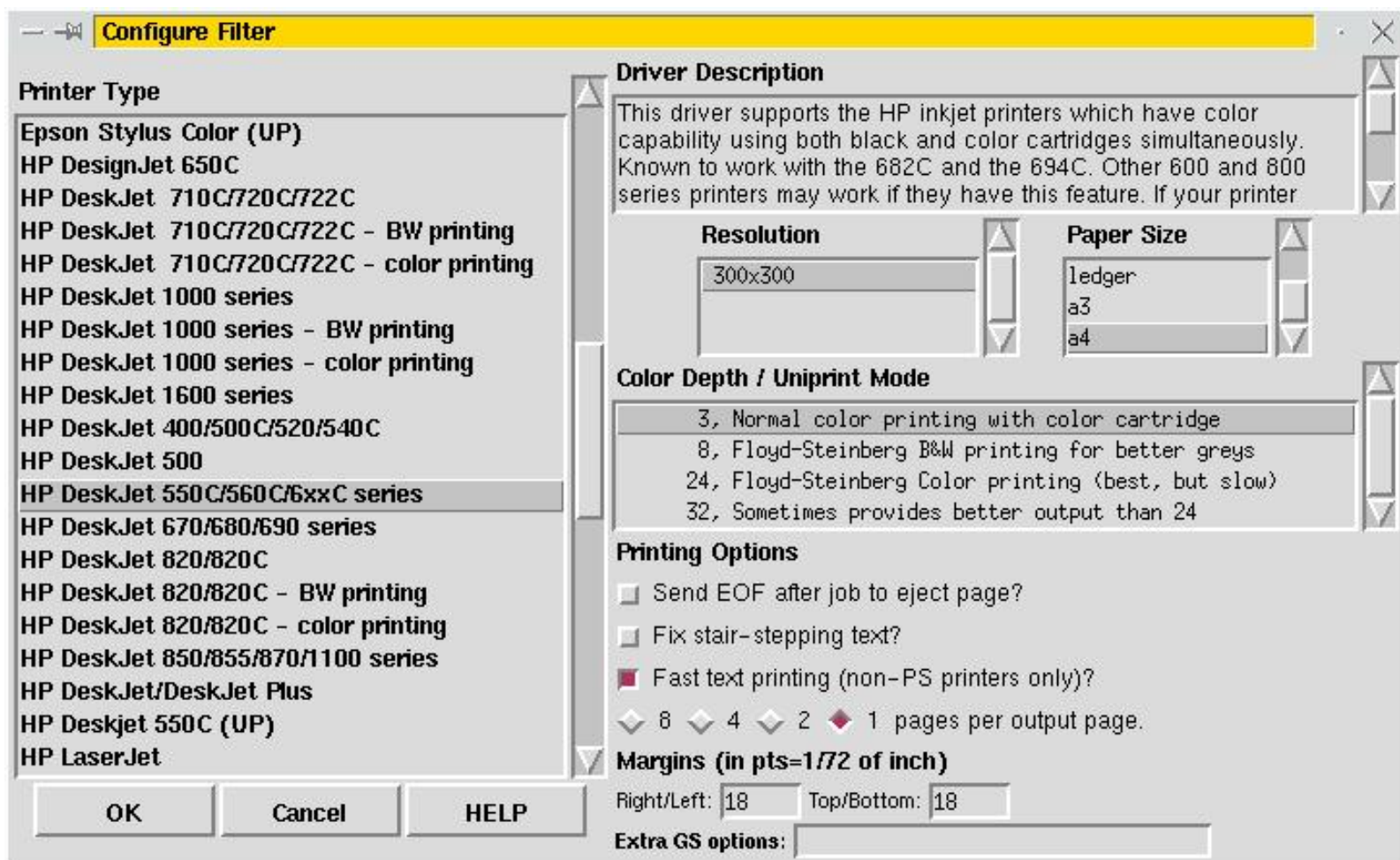
nom de l'imprimante distante (rp=remote printer)

&nbModifications, ajout d'une imprimantesp; **:rp=lp:**

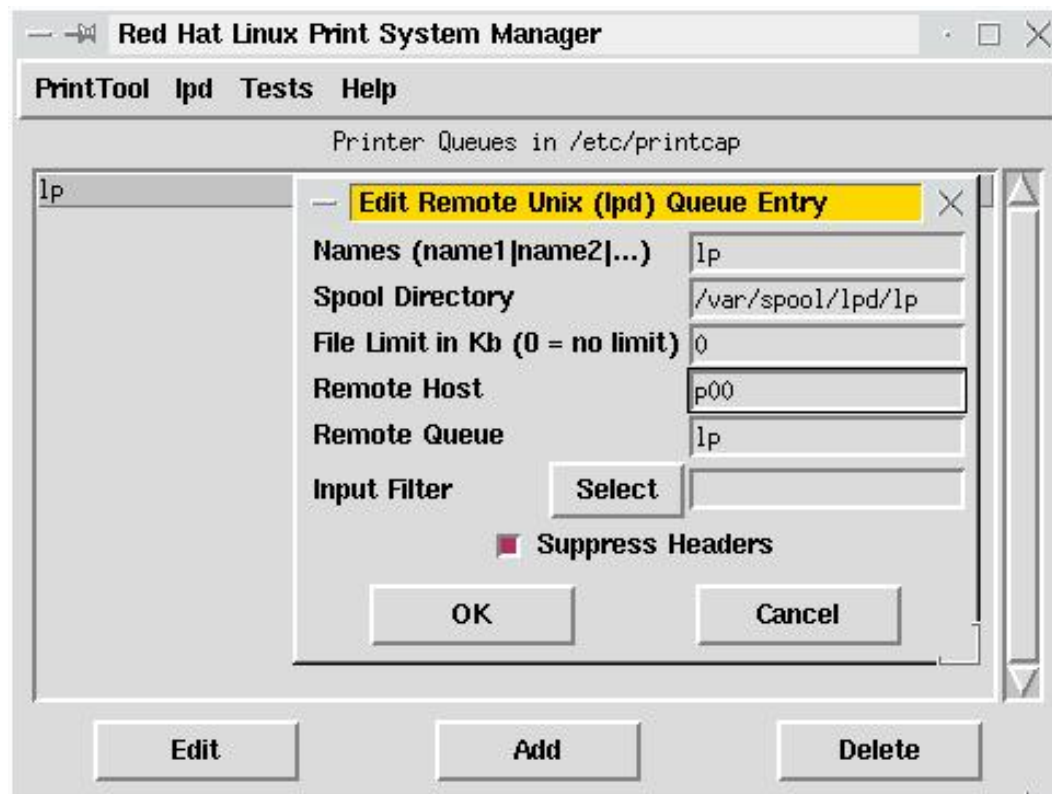
Outil graphique printtool

- On peut complètement modifier le paramétrage du service d'impression, supprimer ajouter des imprimantes grâce à l'utilitaire graphique **printtool**
- Le lancer dans un terminal **X** ou par : *K/appli non KDE/Système/Printtool*, puis **Add**
- Si l'imprimante est connectée à un port parallèle, elle est auto-détectée, en général sur **/dev/lp0**, l'équivalent de LPT1.
- Normalement, l'installation va créer un rép. de spool dans **/var/spool/lpd/** qui appartient à **root** avec les droits 755
- Exemple d'installation d'une imprimante locale





- Exemple d'installation d'une imprimante lointaine



Commandes générales d'impression

Requête d'impression : **lpr** <liste fichiers>

Sur la station Linux cliente du service (pour une imprimante distante ou locale) la commande

lpr /etc/smb.conf demande l'impression du fichier spécifié en argument, dans la file d'attente.

Par défaut, il s'agit de la file **lp**, c'est-à-dire le fichier situé à **/var/spool/lpd/lp:**.

Remarques complémentaires

- Pour imprimer sur une autre imprimante : **lpr -P<nom printer>**
- Si un utilisateur envoie régulièrement ses fichiers à une imprimante particulière, on peut affecter la variable d'environnement **PRINTER** dans son profil personnel, le fichier **\$HOME/bash_profile**
- Syntaxe générale : **lpr -P<nom-printer> -#<nb copies> <liste fichiers>**
- **lpr seul**, attend en scutant l'entrée standard; donc si on saisit du texte, terminé par le caractère de fin de fichier **^D**, on l'imprime directement !

Suivi des jobs d'impression

- **lpq** affiche les travaux de la file d'attente de l'imprimante **lp**
- **lprm** permet de supprimer un job d'impression en attente..
- Naturellement, seul **root** peut supprimer des travaux dont il n'est pas le demandeur.

Cette gestion est plus facile avec un outil fonctionnant sous X.

Sur le serveur, en session X-KDE, on peut lancer l'utilitaire **KLpq** par la commande :

K / Utilitaires / File d'impression

Il permet de lister, supprimer des jobs en attente ou d'en modifier les priorités.

Compléments

- Utilitaire d'administration en ligne de commandes **lpc** : voir le manuel `man lpc`
- Pour imprimer sous [SAMBA](#)
- Examiner le script **lpd**



Introduction au serveur SAMBA

*ou comment transformer une station sous Windows 9x en client Linux ...
en lui faisant croire qu'elle se trouve sur un réseau NT !*

Qu'est-ce que SAMBA ?

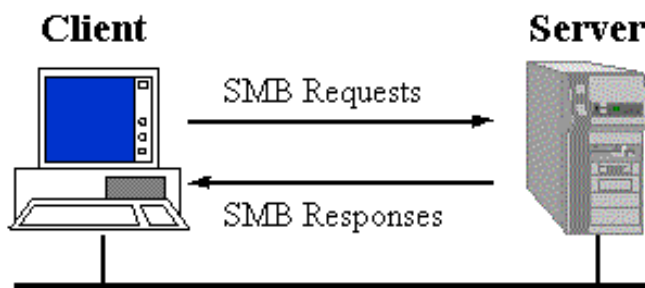
SaMBa est un ensemble de programmes qui permettent de connecter à un serveur LINUX, des stations fonctionnant sous des systèmes divers : Windows 3.11, Windows 9x, Windows Nt, OS/2, Mac....

Le serveur Linux est en mesure de se conduire comme un serveur de fichiers capables d'offrir les services habituels sur un réseau :

- partage de fichiers et de répertoires,
- partage d'imprimantes,
- respect des comptes utilisateurs
- gestion des permissions d'accès
- exécution de scripts de connexion personnalisés

Le protocole de communication sous-jacent qui permet cette communication Linux-Dos/Win s'appuie sur **NetBios** et s'appelle **smb=Server Message Block**.

- Il s'agit en fait de l'implémentation sur Unix, donc Linux d'une émulation d'un serveur **LanManager**, développé par Microsoft vers 1987, à partir d'une création d'IBM (1985).
- Le projet Samba a été initié dès 1991 puis développé par un australien, Andrew Tridgell. Celui-ci lui donna ce nom, en choisissant un nom voisin de SMB en interrogeant un dictionnaire Unix, par la commande `grep "^s.*m.*b" /usr/dict/words`
- Son fonctionnement est conforme au schéma client-serveur



Applications		
SMB		
NetBIOS		IPX/SPX
TCP/IP	NetBEUI	
PPP, 802.x ...		
Media: Token Ring, Ethernet, Serial		

- Le serveur offre ses ressources (système de fichiers, imprimantes ...) aux clients Windows qui s'y connecteront sous un compte créé par `root`, après une authentification par mot de passe.
Le travail est partagé par 2 "daemons" : **smbd** pour le service serveur et **nmbd** pour le service de résolution des noms Netbios.
- Du côté client, le protocole SMB fonctionne au-dessus de plusieurs protocoles.
Il nécessite NetBIOS au dessus de TCP/IP (par contre NetBEUI n'est pas utile)
- Chaque demande de connexion par Samba, d'une station au serveur Linux, laisse une trace stockée dans un fichier `log.%m`, situé dans le répertoire `/var/log/samba` (%m désigne le nom de la station).
Toute connexion pourra donc être identifiée de manière précise puis examinée sur une ligne de ce fichier :
nom d'utilisateur, nom de la machine, date, heure de début, heure de fin, services utilisés...
- Samba peut implémenter la sécurité au niveau de l'utilisateur, (ce qui est recommandé) et non au niveau des ressources comme c'est le cas dans les réseaux de type WorkGroup.

Configuration de SAMBA

Sur les stations DOS-Windows

Paramétrage

Samba ne permet d'accéder à la station Windows qu'à travers le protocole TCP/IP, ce qui semble bien normal .

Il a besoin aussi du protocole **NETBIOS**

Sur chaque machine cliente, il faut tout d'abord ajouter TCP/IP et NETBIOS si ces protocoles sont absents. Bien vérifier que NetBios est activé avec TCP/IP (Voisinage réseau/Propriétés TCP/IP, onglet NetBios).

Il faut affecter une adresse IP à chaque station dans le même sous-réseau que le serveur Samba-Linux.

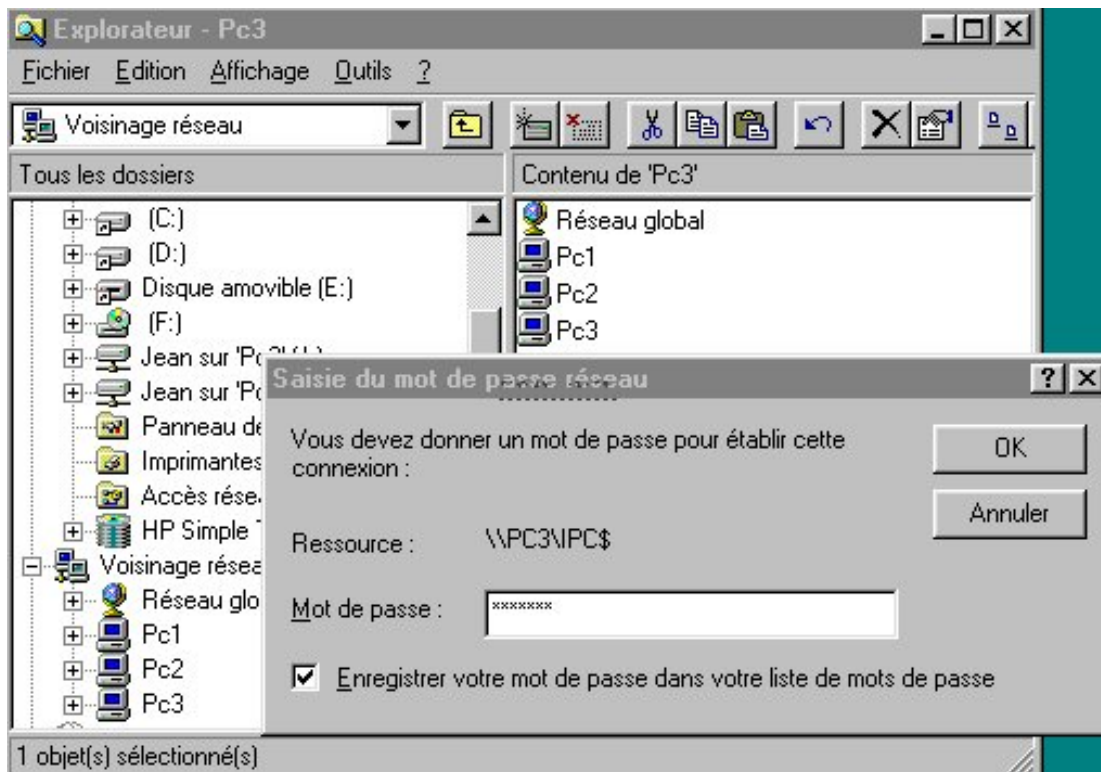
Par exemple, on peut choisir d'affecter une adresse IP statique en établissant un plan d'adressage :

li>l'adresse de sous-réseau **10.177.200.10x**, **x=1,2,3 ..**

- les noms Microsoft des stations : **PCx par ex : PC1 .. PC8**
- le nom de groupe de travail (au sens **workGroup**) qui sera donné dans smb.conf : **fctice77**

Après l'inévitable redémarrage, et l'exigence d'accéder au CD Windows, l'utilisateur devrait :

- "voir" le serveur p00 directement dans son "voisinage réseau".
- Si le fichier smb.conf a été convenablement configuré, le nom du serveur doit apparaître dans le groupe fctice77, sinon il devrait apparaître dans MyGroup (qui est le nom par défaut du Workgroup dans smb.conf)
- Si l'utilisateur stagex s'est connecté sur le réseau Windows sous un compte (login: stagex/ password: stgx) créé sur le serveur Linux, alors il pourra immédiatement avoir accès à son répertoire personnel /home/stagex, qui apparaît sous l'aspect d'un dossier qui porte son nom de connexion.
- Si le mot de passe du compte Linux n'a pas été donné à la connexion proposée par le client réseau Microsoft, il sera alors exigé, comme le montre l'écran ci-dessous :



Installer l'imprimante Linux sur les stations

- On suppose l'imprimante déjà installé sur le serveur Linux et déclarée sur le serveur Samba (voir configuration dans `/etc/smb.conf`).
- Lancer l'Assistant d'ajout d'imprimante (Paramètres /imprimantes)
- Choisir imprimante réseau
- Parcourir le *voisinage réseau* pour détecter l'imprimante : par exemple, choix de `lp` sur **p00**, le serveur Linux
- Le nom de la file d'attente serait alors `\\p00\lp`
- Choisir le modèle d'imprimante et le nom sous lequel elle apparaîtra sur la station cliente (par ex Olivetti sur PC1-Linux).
- Pour vérifier faire imprimer une page de test.
- Désormais, l'imprimante partagée sera visible dans le voisinage réseau au même titre que le rép. personnel de l'utilisateur connecté et des divers dossiers et périphériques partagés.

Remarque : ainsi , une même station peut imprimer sur une imprimante Linux, et une imprimante Windows.

Sur le serveur Linux

Installation

Il est recommandé de demander sa mise en place lors de l'installation.

Sinon il faut monter le cdrom : `mount /dev/cdrom`, chercher le package **samba** et l'installer par :

`rpm -ivh samba-*`

Tous les paramétrages sont ensuite effectués dans un seul fichier : **`/etc/smb.conf`**

Paramétrage de smb.conf

- Faire une sauvegarde de ce fichier par prudence (`cp smb.conf smb.old`) puis l'éditer
- Ce fichier est organisé en sections (à la manière des fichiers ini de Win 3.x).
- Les 2 principales sections, prédéfinies sont **[global]** et **[homes]**.
- L'administrateur root peut éditer, modifier et ajouter des sections, pour définir de nouvelles ressources à partager
- De façon générale les permissions de partage définies dans ces sections ne peuvent pas dépasser les permissions des fichiers du serveur hôte.
- La doc la plus complète se trouve dans le manuel `man smb.conf`

Vérifier et activer les changements

1. Important : lancer l'utilitaire **testparm** permet de tester la syntaxe du fichier de configuration et de déceler les erreurs. Il est recommandé de le lancer systématiquement lors de la mise au point de `smb.conf`. Il diagnostique des erreurs de syntaxe et des incohérences dans les choix des clauses.
2. Ne pas oublier à chaque changement effectué dans `smb.conf` à relancer les processus ! Si les processus "démons" sont actifs, il faut les arrêter puis les relancer par :

```
# conseillé plutôt que la commande globale smb restart
/etc/rc.d/init.d/smb stop
/etc/rc.d/init.d/smb start
```

3. L'écriture d'un petit script shell est le bienvenu pour cette relance ...
Le placer dans un répertoire inclus dans le **PATH**.
Par exemple :

```

vi /usr/sbin/smb
#!/bin/bash
if [ $# = 0 ]
then
echo "Argument : stop | start | restart | status"
exit 1
else
/etc/rc.d/init.d/smb "$1"
fi
:wq (pour sauvegarder)
chmod 700 /usr/sbin/smb (droit d'exécution pour root)

```

Les principaux paramètres

paramètre	valeur par défaut	description
path =		chemin du rep à partager
comment =		texte visible dans le voisinage réseau client
guest ok = yes no (ancien nom : public)	no	partage en accès libre sans authentification
valid users =	tous	liste des users autorisés à se connecter à la ressource
printable = true false	false	partage d'un service d'impression et non de rép.
writable = yes no	no	permet ou non l'écriture sur le rép., contraire de read only
write list =	tous les utilisateurs	liste des users autorisés à écrire
browseable =	yes	visibilité du partage par tous, <i>même les users non autorisés</i>
create mode mask =	0744	droits maxi accordés à un fichier créé dans la ressource ces droits seront en intersection (and) avec les droits Linux (umask)
directory mode mask =	0755	droits maxi accordés à un répertoire créé dans la ressource ces droits seront en intersection (and) avec les droits Linux (umask)
force directory mode =	000	droits imposés lors de la création du rép. composé par un opérateur OR avec les droits usuels
force group =		Impose un groupe propriétaire d'un fichier lors de sa création dans le partage
hide dot files =	yes	cache les fichiers cachés au sens Linux, commençant par un point
hosts allow hosts deny =	toutes les stations aucune	ressource réservée interdite à la liste des stations (adresses IP)
max connections =	0	nb de connexions à la ressource illimité, sinon maxi

La section globale

[global]

```

# donner le même nom de groupe de travail
que celui des stations Windows 95/98 (Voisinage réseau/identification)
workgroup = FCTICE77

# compte à utiliser pour les accès invités aux partages
guest account = nobody ;

# accès multi utilisateur
share modes = yes ;

# restreindre par sécurité les sous-réseaux autorisés à se connecter au serveur
# ici on se limite aux adresses réseau privé 10.194.2.0
et à l'interface "loopback"
hosts allow = 10.177.200. 127.
# on peut exclure des machines de l'accès au réseau
hosts allow = 10.177.200. EXCEPT 10.177.200.125
# d'autres possibilités existent : voir le manuel man smb

# indique l'adresse IP de l'adaptateur du serveur et le masque de sous réseau
interfaces = 10.177.200.110/255.255.255.0

# indique l'emplacement du fichier printcap, récapitulant
toutes les imprimantes installées sur le serveur Linux
printcap = /etc/printcap
# partage toutes les imprimantes définies dans le fichier printcap
load printers = yes

# utiliser un fichier de trace pour chaque machine qui se connecte
log file = /var/log/samba/log.%m

# choisir le mode de sécurité : user ou share
security = user

```

Le répertoire personnel

[homes]

```

# accès au rép. personnel de chaque utilisateur.
# la valeur du champ comment apparaîtra dans le voisinage réseau
# inutile pour cette section de préciser le path, c'est celui de l'utilisateur, en fait /home/%u
comment =Répertoire personnel
browsable = no
writable = yes
create mode = 0700

```

Rendre un répertoire public

L'objectif est de rendre un répertoire partagé totalement (lecture/écriture) à tous les users

D'abord, le créer ou vérifier qu'il existe.

En règle générale, le gestionnaire le crée dans le répertoire **/home/**, lieu de regroupement des répertoires personnels:

```
mkdir /home/tmp
```

```
ls -l
```

renvoie les droits par défaut **drwxr-xr-x**

```
chmod 777 public
```

pour y ajouter les permissions d'accès et d'écriture pour tous

Pour permettre le partage de ce répertoire commun `/home/tmp`, il suffit de modifier la section **[public]** déjà présente et d'enlever les symboles ; pour dé-commenter les lignes en ajoutant des commentaires.

[public]

*# Ce répertoire aura donc pour nom de partage " public "([public]),
la valeur du champ comment apparaîtra dans le voisinage réseau*

Le répertoire à partager est /home/samba
comment =**Répertoire public**
path = **/home/tmp**

il pourra être accessible par tous les utilisateurs
public = **yes**

il est accessible en écriture
writeable = **yes**
les fichiers créés sont en lecture seule, sauf pour le propriétaire
create mode = **0755**

Partager un répertoire pour un groupe

Il s'agit ici de configurer un partage de répertoire pour un groupe.
Dans la section **[HOMES]** sont définis l'accès au rép. personnel de chaque utilisateur.
Pour permettre le partage d'un rép commun **/home/rep-stagiaire**, les lignes suivantes doivent être rajoutées :

[stagiaire] *# Ce répertoire aura donc pour nom de partage stagiaire*
comment =Partage pour le groupe stagiaire exclusivement

Le répertoire à partager est /home/partage
path = **/home/rep-stagiaire**

il ne pourra pas être accessible par tous les utilisateurs
public = **no**

liste des utilisateurs autorisés (avec ou sans virgule)
valid users = **stage1 stage2 stage3 ...**
ou mieux, indication du groupe autorisé
valid users = **@stagiaire jean**

on pourra y écrire (bien sûr par ceux qui peuvent y accéder..)
writeable = **yes**

les permissions par défaut des fichiers créés (le mot mode peut être remplacé par mask
create mode = **0640**

Partager des applications

[logiciels]

comment = Applications partagées sur le serveur
root doit créer ce répertoire et déléguer sa gestion à un groupe d'utilisateurs. # Dans la suite, ce groupe sera appelé
admin (contenant au moins l'utilisateur **admin/admin**) des droits de propriété et permissions
path = **/appli**
public = **yes**
le rép. ne doit pas être en lecture seule pour tous
writeable =**no**
le groupe admin peut seul installer les applications
write list = **@admin**

Partager le lecteur de cd-rom

On crée dans le fichier `smb.conf` une section `cdrom` et on indique le chemin d'accès **path = /mnt/cdrom**. Bien sûr la présence d'un Cd n'est pas suffisante, il doit être monté sur le serveur !

```
[cdrom]
# chemin d'accès au pseudo-répertoire de montage du CD
path = /mnt/cdrom
# accessible à tous les utilisateurs
public = yes
# l'écriture sera interdite
writeable = no
```



Exercice

- Supposons que le serveur Linux possède un lecteur ZIP dont le pilote est installé et dont le point de montage est `/mnt/zip`.
Ecrire la section **[zip]** définissant le partage de cette ressource en lecture pour tous, et en écriture seulement pour `jean`.
- Ecrire la section **[web]** permettant (seulement) à un user (login=`webadmin`/mot de passe=`apache`) d'administrer le site web, à partir d'une station Win9x quelconque.

[Réponses](#)

Paramétrage avec Linuxconf 1.16

Config réseau / tâches serveur / serveur de fichier samba /

```
Administration de Samba /
  Conf par défaut [global]
  Conf par défaut pour un rep utilisateur [homes]
  Partage de disques
    tmp      Rép. temporaire
    public   Rép. public
    stagiaire ...
```

Station Linux, cliente d'un serveur Windows9x

Un client SMB pour Linux est aussi inclus dans Samba.

On peut alors à l'inverse connecter un client Linux à un "serveur" Windows. Le client pourra alors monter une ressource Windows déclarée partagée.

Montage de répertoire Windows

- La commande **smbmount** permet de monter (comme par NFS) des répertoires distants d'une machine Windows9x, sur l'arborescence Linux, et ainsi de disposer des fichiers.
- Pour monter la ressource `//PC1/C_PC1` sur le répertoire `/mnt/disk_pc1` du système de fichiers :
smbmount //PC1/C_PC1 /mnt/disk_pc1
Password : valider.
On parcourt ensuite le répertoire `/mnt/disk_pc1` comme un dossier local.
- S'il y a blocage (faire `ctrl-C`) et se demander :
Le client Samba connaît-il la machine nommée PC1 ?
--> il est nécessaire que le nom de l'hôte (ici PC1) ait été déclaré dans `/etc/hosts`

La ressource C_PC1 est-elle déclarée partagée sur le serveur Windows9x ?

--> clic-droit dans l'explorateur sur le disque concerné/partager ../ etc..

- Le client Samba respecte le type de partage Windows en lecture seule ou lecture/écriture. Supposons que C:\temp soit un sous-répertoire du disque C: du serveur WorkGroup PC1
C: est partagé en lecture seule sous le nom de partage C_PC1
C:\temp est partagé en lecture/écriture sous le nom de partage C_temp_PC1

Monter les ressources :

smbmount //PC1/C_PC1 /mnt/disk_pc1

smbmount //PC1/C_temp_PC1 /mnt/temp_pc1

Tester :

Transférer des fichiers dans C:\temp

Par exemple, copier /etc/fstab dans /mnt/disk_pc1 et dans /mnt/temp_pc1.

- Pour partager les fichiers montés en lecture-écriture pour tous, on ajoute le paramètre **-f 777**
- Pour démonter la ressource Windows, utiliser `subumount` point-montage :
smbumount /mnt/disk_pc1

Problèmes rencontrés

Pas de visibilité du serveur dans le "voisinage réseau" de la station Win9x

- Sur le serveur, les 2 démons `smbd` et `nmbd` sont-ils à l'affus ?
---> vérifier avec la commande `/etc/rc.d/init.d/smb status`
---> si cela n'est pas encore fait, activer le service **smb** au démarrage avec l'utilitaire `ntsysv`
- Sur la station cliente, le protocole *NetBios* est-il bien activé au-dessus de TCP/IP ?

Visibilité du serveur SMB, mais erreur à la connexion

- Problème de connexion sous Windows9x
Une station 98 correctement configurée (domaine MS, NetBios activé) voit le serveur SMB, mais le processus de connexion n'aboutit pas ...
Le problème vient de MS qui a modifié l'authentification des mots de passe. Windows98 les envoie maintenant cryptés alors que par défaut le serveur Samba les attend en clair ..
Il faut donc instaurer une cohérence, soit activer ce cryptage (recommandé, voir [chapitre suivant](#)), soit intervenir sur la base de registres des stations pour inhiber le cryptage.

Marche à suivre pour ne pas crypter les mots de passe

1. Lancer `c:\windows\regedit.exe`
2. Dans la base de registre, se placer dans le rép.
`HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\VxD\VNETSUP`
3. Menu Edition / Nouveau / valeur DWORD
4. Une nouvelle entrée est créée, lui donner le nom *EnablePlainTextPassword*.
5. Double-cliquer pour l'éditer, et donner la valeur **hexadécimale 1**.
6. Fermer et redémarrez la machine .. et ça fonctionne !
7. On peut aussi exécuter un fichier **.reg** qui contient
REGEDIT4

```
[HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\VxD\VNETSUP]
"EnablePlainTextPassword"=dword:00000001
```

- "Serveur pxx non accessible, le réseau est occupé"
Ce message d'erreur a été rencontré sans raisons évidentes.
Bizarrement (?) la connexion a quelquefois été rétablie :

- en commentant la ligne **hosts allow = 10.194.2.**

- en activant la ligne **interfaces= 10.194.2.100/24** (n° ip du serveur, /24 indiquant le masque de sous-réseau)

Difficulté de résolution des noms Netbios

- Le serveur Samba doit pouvoir "résoudre", c'est-à-dire trouver le N° IP correspondant à un nom NetBIOS envoyé par une station, en quête d'un serveur ...
- Lorsqu'on paramètre une station, en effet, on lui donne un nom personnel et un nom de workgroup, pas de nom ni d'adresse IP de serveur ! Il faut aider Samba ! Si on indique rien la recherche s'effectue par **broadcast** dans le même sous-réseau et apparemment cela ne fonctionne pas toujours bien.
- Il est préférable d'indiquer au serveur, dans `/etc/smb.conf`, une méthode explicite de résolution.
 1. La méthode "host"
Le serveur utilise le nom NetBIOS comme nom Linux, et fait appel à un serveur DNS. Par défaut, il s'agit du fichier **/etc/hosts**.
En cas de problème de connexion d'une station, aller voir et si nécessaire compléter ce fichier.
 2. La méthode "wins"
 - C'est celle qui semble recommandée ...
Le serveur Samba est configuré pour être aussi **serveur wins**. Il enregistre alors les correspondances : **nom netbios <- -> adresse IP**
 - Sur chaque station de travail, dans voisinage réseau renseigner l'onglet *Configuration Wins* en indiquant l'adresse IP du serveur.
 - Dans `smb.conf`, section [global], décommenter la ligne
wins support = yes
SANS modifier la ligne `wins server = w.x.y.z`

Compléments et références

Références

- documentation dans `/usr/doc/samba-2.0.5/docs/textdocs/`, en particulier `DIAGNOSIS.txt` et `ENCRYPTION.txt`
- pour les courageux, voir le **man samba.conf**
- Le livre *Samba, installation et mise en oeuvre*, édition O'Reilly (parution août 2000)
- [La "gazette" Linux](#)
- <http://fr.samba.org/samba/samba.html>
- <http://samba.linuxbe.org/>



TP Administration par WEBMIN

Connexion au service Webmin

- Un utilisateur autorisé se connecte par protocole http au port TCP 10000 par l'URL **http://p00:10000**
- La page d'accueil d'accueil se trouve réellement à /usr/share/webmin/index.cgi
C'est la racine du site de webmin.

Webmin

Version 0.80 sur p00.maison.fr (Mandrake Linux 7.0)

[Site internet de Webmin](#)
[Ecrire à l'auteur](#)

Webmin

Système

Serveurs

Matériel

Autres



[Aide de Webmin](#)



[Configuration de Webmin](#)



[Serveurs Webmin](#)



[Utilisateurs Webmin](#)

[Changer d'utilisateur](#)

Paramétrage

Choix de la langue

onglet *Webmin/Webmin Configuration*

lien *Langue*, choisir .. et activer le changement

Choix des stations autorisées

onglet *Webmin/Webmin Configuration*

lien *contrôle d'accès par adresse IP*

n'autoriser que qq machines à être des lieux de connexion

Ajouter un nouvel utilisateur webmin

- onglet *Webmin/Webmin Configuration*
- lien *utilisateur webmin*
- en bas, lien vers *Créer un nouvel utilisateur Webmin*
- par exemple, autoriser un utilisateur **admin** (mot de passe **admin**), déjà créé sous Linux, à devenir gestionnaire du module *Utilisateurs et groupes*

Gestion des utilisateurs par admin

admin crée un utilisateur

- sur la page d'accueil, lien *changer d'utilisateur*
- se connecter comme **admin/admin**

- celui-ci constate qu'il n'a naturellement accès qu'à un seul module
- **admin** crée un utilisateur, par exemple **toto/toto**, de répertoire personnel **/home/toto** (le mot de passe est donné en clair et sera crypté dans **/etc/shadow**)

admin crée le groupe eleves

- lien *Créer un groupe*
- nom du groupe : **eleves**, GID : 600, puis *créer*
- vérifier que le groupe **eleves** est entré dans la liste des groupes

admin crée plusieurs utilisateurs

- lien *Créer plusieurs utilisateurs*
- il faut donner le nom d'un fichier texte contenant le descriptif des comptes à créer.
Ce fichier peut se trouver aussi bien sur le poste d'administration que directement sur le serveur
(attention, le format des fichiers texte sous DOS et sous Linux ne sont pas identiques, les caractères de fin de ligne étant différents)
- créer le fichier **/home/tmp/users.txt** au format adéquat avec **mc** par exemple.
Celui-ci doit contenir des lignes successives de la forme (les champs soulignés sont obligatoires):
nom:pass:uid:gid:vrainom:repertoireperso:shell:min:max:avertissement:inactif:expire
- exemple
eleve1:ele1::600::/home/eleve1:/bin/bash:::::
eleve2:ele2::600::/home/eleve2:/bin/bash:::::
eleve3:ele3::600::/home/eleve3:/bin/bash:::::
- vérifier maintenant que le groupe **eleves** (GID=600) contient bien les comptes élèves créés
- vérifier sur le serveur la présence des comptes dans **/etc/passwd** et **/etc/shadow**
- est-il possible de se connecter sur la station sous l'identité d'un des comptes élèves.
Expliquer pourquoi.

Gestion du serveur Samba

- **root** veut accorder maintenant à **admin** la gestion des comptes Samba
Que doit-il faire ?
- **admin** se connecte à **webmin** et examine le paramétrage de **smb.conf**
Retrouver dans les pages *Windows Networking, Authentication, ...* l'essentiel des choix de la section [global] de **smb.conf**
- On a vu que par défaut les mots de passe Samba ne sont pas générés.
Pour synchroniser la génération des mots de passe Linux et Samba lors de la création d'un nouvel utilisateur, cocher *Add a Samba user when a Unix user is added* et *Change the Samba user when a Unix user is changed* dans la page *Configure automatic Unix and Samba user synchronisation*
- Créer un nouvel utilisateur sur la station et s'y connecter ensuite sous cette identité.



Suivi et planification des processus

Suivi des processus

- Un processus est un programme en cours d'exécution. Il peut donc y avoir simultanément plusieurs processus du même programme en exécution en mémoire. Le noyau Linux lance, gère les processus et contrôle leur échanges avec les périphériques.
- On sait déjà que le premier processus, ancêtre de tous les autres, est [init](#). Tous les autres sont créés par un processus parent et appartiennent à un utilisateur. Ainsi à chaque commande, le shell lance un nouveau processus.
- Pour lancer un processus en tâche de fond (en arrière-plan), faire suivre la commande du symbole **&**. Ainsi, dans une console où on a lancé **startx &**, on peut reprendre la main et lancer une autre commande.

Pour connaître tous les processus en cours de fonctionnement

- **ps** : liste les processus actifs lancés dans la console courante.
- **ps aux** : affiche la liste de tous les processus, avec leur numéro PID, le terminal tty où ils ont été lancés (sinon ?).
- **pstree** permet de visualiser la filiation des processus sous forme arborescente.

Pour supprimer un processus

Si le processus est bloqué, il faut d'abord connaître le numéro PID du processus qui ne répond plus, pour pouvoir le "tuer"

- **kill PID**
- **kill -9 PID** action encore plus radicale !

Exemple : déconnecter radicalement toto

```
ps aux | grep toto
---> toto 858 ..... -bash
kill -9 858
```

Connaître l'état de la mémoire

La commande **free** affiche la mémoire disponible, utilisée, libre ...

Connaître les ressources utilisées par les processus

- La commande **top** affiche une page d'information, périodiquement mise à jour (taper q pour quitter), pour gérer les processus et être informé de la charge de travail du CPU et de l'utilisation mémoire.
- L'équivalent graphique existe sous X-KDE, lancer K/système/gestionnaire de tâches **Ktop**



Manipulations

Connexion comme stagex dans tty1 et dans tty2

Lancer mc dans tty1, afficher un fichier

Dans tty2, **ps aux**, repérer le numéro **PID** du programme mc le supprimer par **kill PID**, et vérifier le résultat.

Crontab : planification des tâches

- **Cron** est un service qui exécute toutes tâches (commandes et scripts) pour les utilisateurs. (Cron vient de chronos, le dieu du temps)
Il est possible de planifier ces tâches à l'avance et même de les faire lancer périodiquement. On peut obtenir l'émission de messages de compte-rendu de ces tâches.

- La commande **/usr/bin/crontab** permet cette programmation. Son usage est en principe réservée à **root**. On peut toutefois autoriser certains utilisateurs. Pour cela on en dresse la liste sur des lignes successives dans le fichier **/etc/crond.allow**, de façon symétrique, on peut mettre dans **/etc/crond.deny** la liste des utilisateurs non autorisés.
- Fonctionnement
 - Le processus **crond** est normalement lancé au démarrage.
On peut le lancer ou l'arrêter avec **/etc/rc.d/init.d/crond**
 - Il lit toutes les minutes le fichier **/etc/crontab** et les fichiers présents dans le répertoire **/var/spool/cron** pour voir si des tâches doivent être exécutées.
 - Chaque action de **crond** ajoute une ligne de message dans le fichier **/var/log/cron**
- Modifier le fichier **/etc/crontab**
Commande **crontab [-u user] {-l | -r | -e }**.
Options :
 - **crontab -l** affiche le fichier crontab de l'utilisateur
 - **crontab -r** efface ce fichier
 - **crontab -l -u jean root** examine le fichier crontab de l'user jean
 - **crontab -e** crée ou édite (pour modification) un fichier temporaire dans **/tmp** ouvert dans **vi**
Lors de la sauvegarde, le fichier est écrit dans **/var/spool/cron/\$USER**, où **\$USER** est le nom de login de l'utilisateur.
- Syntaxe des lignes de **/etc/crontab**
Chaque ligne du fichier contient 6 champs, les 5 premières déterminent les moments d'exécution de la tâche décrite au 6ème champ.
 - les 5 premiers, séparés par des espaces, décrivent la périodicité :
minutes (0-59), heures (0-23), jour du mois (1-31), mois de l'année (1-12), jour de la semaine (0-6, 0= dimanche)
 - le 6ème est la commande à exécuter, ce peut être naturellement un script qqc
 - un champ temporel peut contenir :
 - une valeur précise et valide pour le champ (par exemple 15 sur le champ minute)
 - une liste de valeurs valides, séparées par des virgules (1,3,5 dans le champ mois : janvier, mars, mai)
 - un intervalle valide (1-5 dans le champ jour : du lundi au vendredi)
 - * pour signifier toutes les valeurs possibles du champ (* dans le champ minute : toutes les minutes)
 - */5 (dans le champ minutes : tous les 5 minutes), 1-24/3 (dans le champ heures : toutes les 3 heures)
- Exemples :

```
# exécution chaque 1er et 15 de chaque mois à minuit
0 0 1,15 * * commande
# exécution toutes les heures passées 15 minutes
15 * * * * commande
# exécution tous les matins du lundi au vendredi à 7 h 30
30 7 * * 1-5 commande
# exécution tous les quarts d'heure de 15 à 19h du lundi au vendredi
# seulement en 1ère quinzaine du troisième trimestre
0,15,30,45 15-19 1-15 7-9 1-5 commande
# trouver puis nettoyer le répertoire /tmp des vieux fichiers (non modifiés
# depuis 31 jours) tous les 1er jour de chaque mois, à 2 heures du matin
0 2 1 * * find /tmp -atime 31 -exec rm -f {} \;
envoyer les messages "Je suis encore au travail !" tous les quarts
d'heure à partir de 17 h à tous les utilisateurs actuellement connectés,
et le message "Bonjour chef !" à root tous les jours ouvrables à 10 h
0 10 * * 1-5 write root %Bonjour chef !
*/15 17-20 * * 1-5 wall %Je suis encore au travail !
```



Exercices immédiats

1. Le processus **crond** est-il actif ? (`ntsysv` ou `ps aux | grep cron`)
2. Envoyer toutes les minutes un petit bonjour à l'utilisateur connecté
3. L'utilisateur **toto** ajoute toutes les 5 minutes un message "Bonjour" suivi de la date, dans le fichier `/tmp/bonjour.txt`
4. root fait enregistrer de 9h à 17h, les jours ouvrables :
 - toutes les heures, dans `/var/log/processus.txt`, tous les processus qui tournent sur la machine
 - tous les 5 minutes, dans `/var/log/qui.txt`, tous les utilisateurs connectés

Sauvegarde quotidienne des répertoires personnels

1. Il s'agit d'effectuer une tâche quotidienne de sauvegarde globale des répertoires personnels présents dans **/home** dans un répertoire `var/sauve/` à créer
2. Ecrire la commande d'archivage compressé de `/home/*` dans un fichier `home.tgz` à placer dans `/var/sauve`
3. La sauvegarde doit être quotidienne. A l'aide de la commande `date`, écrire un script permettant l'archivage du 12 nov dans un fichier nommé `home.12nov.tgz`
4. Automatiser cette tâche avec **cron**, à ... 1h du matin

Annexe

Comprendre le fonctionnement des commandes placées dans les répertoires temporels

Fichier **/etc/crontab**

```
# run-parts
01 * * * * root run-parts /etc/cron.hourly
02 4 * * * root run-parts /etc/cron.daily
22 4 * * 0 root run-parts /etc/cron.weekly
42 4 1 * * root run-parts /etc/cron.monthly
```

Fichier **/usr/bin/run-parts**

```
#!/bin/bash # run-parts - concept taken from Debian
```

```
.....
for i in $1/* ; do
    [ -d $i ] && continue
    if [ -x $i ]; then
        $i
    fi
done
exit 0
```

Proposition de corrigés

Toto dit bonjour

```
toto étant un utilisateur autorisé
$ crontab -e
# toto ajoute la ligne (echo -n inhibe le passage à la ligne) :
*/5 * * * * (echo -n Bonjour ! nous sommes le ;date ) >> /tmp/bonjour.txt
```

root surveille ...

```
0 9-17 * * 1-5 (ps aux ; echo "*****" ) >> /var/log/processus.txt
*/15 9-17 * * 1-5 who >> /var/log/qui.txt
```

root sauvegarde les rep. personnels

```
#!/bin/bash
# fichier sauve_home.sh
date=$(date)
set -- $date
tar czvf /var/sauve/home.$3$2$6.tgz /home/*

# extrait de crontab de root (fichier /etc/spool/cron/root)
0 1 * * * /var/home/sauve_home.sh
```



Installation de Linux par le réseau

Situation et objectifs

- Il s'agit d'installer le système Linux via le réseau, sur une machine possédant un adaptateur réseau dont on connaît les caractéristiques.
- Intérêt : installer Linux sur des stations où le lecteur de CD est absent ou ... défaillant.
La version que l'on installe peut être plus récente que celle qui fonctionne sur le serveur.
- On suppose disposer d'un serveur Linux opérationnel sur lequel tourne l'un des 2 services **NFS** ou **FTP**, et qui dispose d'un lecteur de CDROM.
Soient **p00.fctice.fr** son nom complet, **192.168.1.100** son adresse **IP**
- On choisit pour la station, le nom **pc2.fctice.fr** et l'adresse IP **192.168.1.102**

Etapes de l'installation

Préparation de la dk d'installation

- Une disquette de démarrage spéciale est requise. L'image qu'elle doit contenir est **network.img** (pour la distribution Mandrake 7.1).
- Pour cela monter d'abord le CDROM et se placer dans le répertoire images

```
# mount /mnt/cdrom
# cd /mnt/cdrom/images
```
- Insérer une disquette, puis passer la commande de copie :

```
# dd if=network.img of=/dev/fd0
```

Préparation du serveur NFS

- Les services portmap et nfs ont été installés et sont démarrés. Voir si besoin [l'installation et le paramétrage](#)
 Pour vérifier :

```
/etc/rc.d/init.d/portmap status
---> portmap (pid ...) is running
/etc/rc.d/init.d/nfs status
---> rpc.statd (pid ...) is running
---> rpc.mountd (pid ...) is running
---> nfsd ( liste de 8 pid ) is running
---> rpc.rquotad (pid ...) is running
```
- Exporter le point d'accès au cdrom
 Ajouter la ligne suivante dans le fichier d'exportation `/etc/exports`

```
/mnt/cdrom/ pc2(ro)
```

où **pc2** est le nom de la machine à installer.

- Probablement aucun éventuel serveur de noms ne connaît pc2.
Comme le serveur NFS doit connaître la correspondance entre le nom et l'adresse IP, le plus simple est d'ajouter une ligne dans `/etc/hosts` :
`192.168.1.102 pc2.fctice.fr pc2`
- Et bien sûr monter le cdrom et vérifier son accessibilité dans l'arborescence
`mount /mnt/cdrom`
- Vérifier la permission dans le fichier `/etc/hosts.allow`

Sur la station

1. On boote la machine sur la disquette. Après un temps de chargement, un premier écran nous invite à choisir la méthode d'installation entre NFS, FTP et HTTP.
On choisit ici **NFS Image**
2. Il faut désigner le pilote de la carte réseau (ne2000 par exemple) et tenter un "autoprobe", sinon l'adresse IO (0x300 par exemple) et l'irq
3. Paramétrer TCP/IP. Ici on lui affecte l'adresse IP statique prévue : `192.168.1.102` . Le masque complété est `255.255.255.0`. Effacer les 2 autres lignes.
4. Compléter les 2 premières lignes : le nom de domaine `maison.fr` et le nom complet de la machine `pc2.fctice.fr` (attention le clavier est bien sûr à ce stade en QWERTY !)
5. Indiquer la localisation des fichiers à importer par protocole NFS
 - NFS name server : `192.168.1.100` (adresse IP du serveur NFS)
 - Mandrake directory : `/mnt/cdrom` (le point d'accès aux fichiers du CDROM)
6. Normalement, la connexion au serveur NFS devrait démarrer le processus d'installation, et par suite l'installation s'opérer, comme si le CDROM était monté localement.

A étudier

- Tester l'installation de plusieurs stations simultanément par ce procédé
- A comprendre : le rôle que pourrait jouer la disquette d'auto-réplication



Documentation

Auto-documentation

Linux est lui-même très documenté.

A consulter, avant de chercher une aide extérieure ... qui ne fait bien souvent que reprendre de façon incomplète ces informations "de première main".

- La plupart des commandes possède une aide succincte, bien souvent suffisante comme aide-mémoire

Par exemple :

```
$ cp --help | less
```

- Les pages de manuel sont des textes souvent très long et très complet à considérer comme le manuel de référence. Par exemple le manuel de l'utilitaire `lilo` s'examine par

```
$ man lilo
```

- Les HOWTO

Sites WEB généraux

- <http://www.linux-center.org/fr>

Index thématique de pages Web consacrées au système d'exploitation Linux, à ses applications et plus généralement au logiciel libre.

- <http://www.linux-france.com>

- <http://www.linux-kheops.com/doc/>

- <http://www.gnu.org>

le projet GNU, objectifs, textes fondateurs

- <http://www.freenix.org/>

logiciels, faq, howto

- <http://www.linuxbe.org/>

- <http://www.linux.com/>

- <http://www.li.org/>

informations pratiques, nouvelles

- <http://www.cru.fr/linux/>
page Linux du Comité Réseaux Université du CNRS

Sites WEB des quelques distributions

- <http://www.fr.redhat.com/> (fr)
- <http://www.linux-mandrake.com> (fr)
- <http://www.suse.com> (fr)

Sites WEB développement

- <http://www.kde.org>
- <http://www.gnome.org>

Associations

- <http://www.aful.org>
- <http://www.april.org>

Sites divers

- <http://www.cru.fr/free/> *"Les logiciels libres au CRU"*
La cellule technique du CRU utilise les logiciels libres à chaque fois que cela est possible
- <http://www.independance.seul.org> *"Linux pour les masses"*
- <http://www.linuxgirls.org> *"Linux au féminin"*

HOW-TO / FAQ

<http://www.freenix.org/linux/HOWTO/>

Listes de diffusion

- linux-educ@listes.ac-creteil.fr
 - liste de discussion de l'académie de Créteil pour la promotion de Linux dans l'enseignement
 - accès WEB (abonnement, archives) <http://listes.ac-creteil.fr>
- samba-edu@tice.ac-caen.fr
 - liste de discussion de l'académie de Caen sur la distribution SambaEdu

- site WEB (abonnement, documentation) <http://www.linux-france.org/prj/edu/sambac/g/>
- linux@nnx.com
 - Liste de discussion générale et de bon niveau
 - Voir les [archives](#)

Livres

- Le système Linux (3ème éd), ouvrage fondamental, éditeur O'Reilly
- Administration réseau sous Linux, éditeur O'Reilly
- Linux , série "Grand livre", Micro-application
- Introduction à Perl, livre fondamental pour découvrir Perl, éditeur O'Reilly
- Logiciels libres (JP Smets-Solanes, B Faucon) Edispher



TCP/IP LINUX

Les outils de base

Les outils suivants sont indispensables à connaître lorsque l'on utilise un système sous Linux. On ne peut ici donner toutes les options de ces commandes. N'oubliez donc pas que l'on peut avoir plus d'aide en tapant la commande suivie de `--help`, mais aussi `man` commande. Exemple **netstat --help** ou **man netstat**. Enfin souvenez vous que sous Linux, on ne peut utiliser indifféremment les majuscules et les minuscules (la commande ping existe, pas la commande Ping).

1. ping

Cette commande est normalement connue de tous. Elle existe dans tous les systèmes. Elle permet de vérifier si une machine distante répond. La syntaxe est des plus simple **ping -c 5 192.168.0.1** pour envoyer 5 pings à la machine dont l'adresse IP est 192.168.0.1.

On peut aussi utiliser le nom de la machine, si celle-ci est renseignée dans votre fichier Hosts ou dans un serveur DNS.

On peut par exemple utiliser ping pour vérifier si la connexion est toujours active ou pour la monter.

Si vous ne placez pas l'option `-c 5` pour n'envoyer que 5 pings, la commande ne s'arrête pas. Utilisez alors Ctrl C.

2. ifconfig

ifconfig permet de connaître la configuration de vos cartes réseau, mais aussi de changer celle-ci. Pour changer la configuration de votre carte réseau, vous devez taper

ifconfig eth0 192.168.0.2 netmask 255.255.255.0 broadcast 192.168.0.255

Comme les valeurs que je viens de donner sont standards, vous pouviez simplement taper **ifconfig ETH0 192.168.0.2** (le netmask et broadcast proposés sont ceux correspondant à une adresse de classe C).

Attention au redémarrage de la machine ce changement sera perdu. Il vous faut donc modifier en même temps le fichier `/etc/sysconfig/network-script/ifcfg-eth0`.

Vous pouvez utiliser **linuxconf** pour faire plus simplement le même travail.

On peut aussi désactiver une carte réseau **ifconfig eth0 down** et bien sûr la réactiver **ifconfig eth0 up**

3. arp

La commande arp permet de mettre en correspondance des adresses IP et les adresses MAC. Les options possibles importantes sont

arp -a pour avoir toutes les entrées ARP de la table

arp -d nom_de_la_machine pour supprimer une entrée de la table

arp -s nom_de_la_machine adresses_mac pour ajouter une nouvelle entrée dans la table.

4. route

Cette commande permet de voir, d'ajouter ou d'enlever les routes se trouvant déclarées sur votre machine. Ainsi pour indiquer à votre machine où aller trouver les adresses qui ne sont pas les adresses de votre réseau local, vous devez lui indiquer la passerelle (ou gateway) vers laquelle elle doit envoyer tous les paquets.

Pour voir les routes indiquer **route -n** (on peut aussi utiliser **netstat -nr**) L'option -n permet de ne pas avoir la résolution des noms.

Pour ajouter une route par défaut : **route add default gateway 192.168.0.1** (La passerelle vers qui j'envoie tous les paquets qui ne sont pas pour le réseau local). Pour détruire cette route **route del default**

Pour ajouter une route vers une machine indiquer **route add -host 195.98.246.28 gateway 192.168.0.1** (Indiquer le netmask si celui-ci n'est pas un mask correspondant à la classe de votre adresse).

Pour ajouter une route vers un réseau indiquer **route add -net 195.98.246.0 netmask 255.255.0.0 gateway 192.168.0.1**

Enfin pour supprimer une de ces routes remplacer add par del.

La gateway ou passerelle correspond la plupart du temps à votre routeur.

Pour avoir la route que vous venez d'ajouter à chaque démarrage placer la commande dans le fichier /etc/rc.d/rc.local par exemple.

On peut aussi utiliser **linuxconf** pour faire la même chose.

5. netstat

Voilà une commande moins connue et pourtant très utile. Je ne peux ici commenter toutes les options, je vous propose de lire le **man netstat**. Elle permet en effet de connaître les ports en écoute sur votre machine, sur quelles interfaces, avec quels protocoles de transport (TCP ou UDP), les connexions actives et de connaître les routes.

Pour voir les connexions actives **netstat -nt**, pour les ports ouverts **netstat -ntl**.

On peut aussi vérifier s'il existe une route par défaut, par exemple existe-t-il une route par défaut vers la machine 195.98.246.28 utilisez alors **netstat -nr | grep 195.98.246.28**.

L'option -a énumère les ports en cours d'utilisation ou ceux qui sont écoutés par le serveur.

L'option -i donne des informations sur les interfaces réseau.

6. traceroute

Traceroute permet de déterminer la route prise par un paquet pour atteindre la cible sur internet. On peut utiliser soit l'adresse IP, soit le nom d'hôte. Attention certains FireWall ou routeurs ne se laissent pas voir avec la commande traceroute.

La commande traceroute est très utile pour savoir où peut se trouver un blocage (plutôt ralentissement). Il existe un grand nombre d'options, entre autre il est possible de choisir les gateway (jusqu'à 8) pour atteindre une machine. Je vous conseille donc encore une fois de lire le man traceroute.

7. telnet

Telnet est l'outil indispensable à connaître. Il existe en tant que client sur tous les systèmes. Par

contre Linux dispose en plus d'un serveur telnet permettant d'administrer à distance une machine (quoiqu'il existe maintenant un serveur telnet sur windows 2000). On peut ainsi administrer une machine linux depuis un Microsoft quelconque et, mais cela va de soi, depuis une autre machine linux.

En tant que client, telnet vous permet d'envoyer et de lire vos messages (voir ici).

Pour pouvoir administrer à distance, il faut que le serveur telnet soit installé sur la machine que vous souhaitez administrer. Pensez aussi à vérifier que cela est autorisé dans le fichier `etc/inetd.conf` et dans `/etc/hosts.allow` et `/etc/hosts.deny` (voir tcp wrappers).

Si vous devez vous en servir sur un réseau local ou sur internet, préférez lui **SSH** (ou la version autorisée en France **SSF**), car alors les mots de passe ne se promènent pas en clair sur le réseau. Par défaut il n'est pas possible de se connecter en root avec une connexion telnet. Vous devez utiliser un autre compte et utiliser la commande `su`.

8. ftp

ftp est un outil qui permet de télécharger des fichiers entre machines. Vous connaissez les clients ftp comme `ws_ftp`.

Sous Linux il existe un serveur ftp, que vous activez dans `/etc/inetd.conf`. Il est installé par défaut dans toutes les distributions. Ce serveur ftp n'est pas lié à l'installation d'apache, comme pour les systèmes Microsoft où vous devez installer IIS pour bénéficier de ce service. Attention toutefois le serveur ftp pose un problème de sécurité important, utilisez plutôt **SFTP**, qui est disponible avec **SSH**.

Vous disposez aussi d'un client ftp en ligne de commande sous Linux comme sous Microsoft. La syntaxe étant pratiquement la même.

```
[philippe@lycee1 /]$ ftp localhost
Connected to localhost.
220 lycee1.ac-creteil.fr FTP server (Version wu-2.6.0(1) Mon Feb 28 10:30:36 EST
2000) ready.
Name (localhost:philippe): philippe
331 Password required for philippe.
Password:
230 User philippe logged in.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> binary
200 Type set to I.
ftp> mget *
```

Voici les commandes que vous allez utiliser le plus :

dir	pour lister un répertoire
cd nom_du_répertoire	pour changer de répertoire

get mon_fichier	pour copier un fichier vers votre client (obtenir). Il se place alors dans le répertoire où vous vous trouviez.
mget *	copier tous les fichiers du répertoire vers votre station
put mon_fichier	pour copier un fichier vers le serveur
mput *	pour copier les fichiers se trouvant dans votre répertoire.
binary	pour copier en mode binaire.
exit	pour quitter

Il existe un grand nombre d'autres commandes. Mais vous avez là les principales, pour copier des fichiers entre machines. La commande ftp vous rendra un grand nombre de services, car elle permet assez simplement d'échanger des fichiers entre linux et windows, sans avoir à installer un client ftp ou à configurer samba.

9. nslookup

L'utilitaire nslookup permet d'interroger un serveur de nom (serveur dns) afin d'avoir des informations sur un domaine ou sur une machine. Par défaut nslookup utilise le serveur de nom configuré sur votre machine, vous pouvez toutefois interroger un autre serveur de nom.

```
[root@aleu /]#nslookup
```

Default Server: localhost Car j'ai un serveur dns sur ma machine

Address: 127.0.0.1

>help Pour avoir de l'aide

>**set type = MX** Pour lister les entrées de type MX (à savoir les serveurs SMTP du domaine).

>ac-creteil.fr Le nom du domaine dont vous voulez avoir des MX

Remplacer MX par le type d'enregistrement que vous souhaitez avoir. Par exemple NS pour les serveurs de nom d'un domaine, SOA pour start of authority, PTR pour le reverse, A pour une machine.

Pour avoir toutes les informations

set type=ANY puis le nom du domaine.

On peut aussi utiliser la commande **ls -t CNAME nom_du_domaine** pour avoir tous les enregistrements de type cname (les alias).

Pour interroger un autre serveur DNS que votre serveur par défaut **server NAME 195.98.246.50**.

10. who

Cette commande permet de connaître les personnes qui sont loguées sur votre machine.

11. last

Cette commande vous permet de voir les dernières connexions ayant eu lieu sur votre machine (en fait il lit le fichier /var/log/wtmp).

last sans rien, vous affiche toutes les informations.

last philippe toutes les connexions de l'utilisateur philippe.

last reboot tous les reboot de la machine avec la date.

lastb est une variante de last, dans la mesure ou il ne cherche que les mauvais login (il lit le fichier /var/log/btmp)

12. finger

finger est un service qui vous permet d'obtenir des informations sur les comptes utilisateurs de votre machine. Ce service est à proscrire. _

sa syntaxe est toutefois assez simple **finger toto@la_machine_distante**

Pour avoir plus d'informations utiliser l'option -l

Cet exemple montre les deux connexions ouvertes sur la machine aleu.

```
[root@aleu philippe]# finger -l
```

```
Login: root Name: root
```

```
Directory: /root Shell: /bin/bash
```

```
On since Sun Oct 22 18:34 (CEST) on tty1 39 seconds idle
```

```
(messages off)
```

```
No mail.
```

```
No Plan.
```

```
Login: philippe Name: philippe
```

```
Directory: /home/philippe Shell: /bin/bash
```

```
On since Sun Oct 22 18:48 (CEST) on pts/0 from 10.100.1.19
```

```
No mail.
```

```
No Plan.
```

Par défaut et par sécurité les machines distantes ne permettent pas, ou plus les commandes finger.

13. tcpdump

tcpdump permet de faire des captures de paquets sur votre réseau. Je présente ici tcpdump car on le trouve sur tous les cd des distributions. Il n'est pas le plus agréable à utiliser et des utilitaires de ce type plus conviviaux existent. Mon but n'est pas ici de vous montrer comment devenir un pirate (en capturant les mots de passe qui circulent en clair sur votre réseau), mais plus de vous permettre de vérifier par exemple lorsque votre routeur monte la ligne sans que vous ne soyez capable de donner l'origine de cette montée de ligne. Comme de plus cela arrive la nuit (toujours quand on n'est pas là..!), il peut être utile de placer tcpdump et de capturer les paquets à destination de votre routeur et uniquement cela. Au petit matin en analysant le résultat vous savez quelle machine et quel protocole monte la ligne.

Par exemple pour intercepter tous les paquets vers la machine 10.100.1.5 sur le port telnet

tcpdump -l -q -x host 10.100.1.5 and port telnet

Pour intercepter tous les paquets d'une machine vers une autre sur le port telnet

tcpdump -l -q -x dst 10.100.1.5 and src 10.100.1.19 and port telnet and tcp

Pour avoir tous les paquets qui arrivent sur votre machine 10.100.1.5 ne pas indiquer la source.

14. nmap

nmap est un outil pour scanner les ports ouverts sur une machine distante. Son utilisation est des

plus simple **nmap 192.168.0.1** pour scanner une machine ou **nmap 192.168.0.*** pour scanner les machines se trouvant dans le plan d'adressage 192.168.0.0/24

Utilisez l'option -v pour avoir plus d'informations. On peut bien sur scanner que certains protocoles, par défaut le protocole scanné est TCP. Pour scanner les deux **nmap -v -sU -sT 192.168.0.1**

Les options disponibles sont :

- sT Scanne les ports TCP (attention cela est inscrit dans les fichiers de log de la machine cible).
- sS Est identique au précédent sauf que cela ne laisse pas de trace. (il y a une différence quant à la méthode mais cela n'est pas l'objet de cette présentation).
- sP En fait un ping.
- p 20-140 Ne scanne que les ports entre 20 et 40.
- I Pour avoir plus d'information sur le port ouvert

```
[root@aleu philippe]# nmap -v -sU -sT 10.100.1.1
```

```
Starting nmap V. 2.30BETA17 by fyodor@insecure.org ( www.insecure.org/nmap/ )
```

```
Host (10.100.1.1) appears to be up ... good.
```

```
Initiating TCP connect() scan against (10.100.1.1)
```

```
Adding TCP port 139 (state Open).
```

```
Adding TCP port 25 (state Open).
```

```
Adding TCP port 110 (state Open).
```

```
Adding TCP port 21 (state Open).
```

```
Adding TCP port 135 (state Open).
```

```
Adding TCP port 80 (state Open).
```

```
Adding TCP port 1026 (state Open).
```

```
The TCP connect scan took 1 seconds to scan 1534 ports.
```

```
Initiating FIN, NULL, UDP, or Xmas stealth scan against (10.100.1.1)
```

```
The UDP or stealth FIN/NULL/XMAS scan took 4 seconds to scan 1534 ports.
```

```
Interesting ports on (10.100.1.1):
```

```
Port State Service
```

```
21/tcp open ftp
```

```
25/tcp open smtp
```

```
80/tcp open http
```

```
110/tcp open pop-3
```

```
135/tcp open loc-srv
```

```
137/udp open netbios-ns
```

```
138/udp open netbios-dgm
```

```
139/tcp open netbios-ssn
```

```
1026/tcp open nterm
```

```
Nmap run completed -- 1 IP address (1 host up) scanned in 5 seconds
```

Il n'est pas installé par défaut sur votre machine, mais on commence à le trouver sur le CD d'installation des distributions. Il existe une interface graphique (installer nmap-frontend en version rpm).

Evitez de scanner des machines qui ne sont pas vos machines. Ce produit a pour vocation de

vérifier si votre machine est correctement configurée, pas pour tester les autres.

- **TP 1 :** Modifier de façon définitive à l'aide de la commande `ifconfig` l'adresse IP de votre machine.
 - **TP 2 :** Chercher les informations suivantes :
 - Adresse IP de la machine www.ac-creteil.fr et le nom de la machine web de l'académie.
 - Nom des serveurs SMTP de l'académie de Créteil. Indiquer le premier serveur, ainsi que les serveurs secondaires.
 - Pouvez vous donner tous les CNAME de l'académie ainsi que le nom réel des machines.
 - Pouvez vous indiquer l'adresse IP du serveur proxy de l'académie. Pouvez vous pinguer le proxy.
 - **TP 3 :** Ecrire un petit script qui vous permette de savoir quelle machine de votre réseau envoie des requêtes vers l'internet et sur quel port.
-



TCP Wrappers

1 Pour quoi faire

Le mécanisme de TCP_wrappers permet de contrôler et de restreindre l'accès à certain service réseau. En fait il utilise le daemon tcpd qui intercepte les demandes de connexion à un service et vérifie dans les fichiers hosts.allow et hosts.deny si le client est autorisé à utiliser ce service. Sur les versions de linux actuelles, il est installé par défaut. par contre il n'est pas actif dans sa partie contrôle d'accès.

TCP Wrappers est un élément à mettre en oeuvre pour sécuriser une machine sous linux, il ne peut toutefois pas remplacer complètement un vrai FireWall.

2 Le principe.

Lorsque vous souhaitez vous connecter sur une machine distante en telnet, par exemple, le daemon inetd intercepte votre demande de connexion et vérifie dans le fichier inetd.conf si le service telnet est utilisable. si la réponse est positive, votre demande est passée à tcpd qui vérifie dans les fichiers hosts.allow et hosts.deny si vous avez le droit de vous connecter en telnet, si cela est le cas votre demande de connexion sera autorisée, sinon vous serez rejeté. Dans tous les cas de figure et cela est l'autre fonction de TCP_wrappers tcpd transmettra à syslogd (daemon de log) votre demande (cette demande se retrouvera loguer dans le fichier /var/log/securite).

3 L'installation

Par défaut il est installé avec la plupart des distributions, mais au cas, le paquet à installer est :. tcp_wrappers-x.....rpm.

TCP_wrappers utilise les fichiers suivants : tcpd, inetd, inetd.conf, hosts.allow, hosts.deny, tcpdchk, tcpdmatch.

4 Le fichier inetd.conf

Ce fichier se trouve dans le répertoire /etc. Vous pouvez activer ou désactiver ici des services, en plaçant un # devant la ligne ou en l'enlevant, puis en obligeant la relecture du fichier avec un killall -HUP inetd. Il est possible d'ajouter d'autres services dans ce fichier. Enfin certains l'utilisent sans être dedans par exemple SSH s'il a été compilé avec les bonnes options.

En voici un commenté. On le trouve dans /etc

```

# Version: @(#) /etc/inetd.conf 3.10 05/27/93
# Les premières lignes sont utilisées par inetd
#
#echo      stream      tcp      nowait    root      internal
#echo      dgram       udp      wait      root      internal
#discard   stream      tcp      nowait    root      internal
#discard   dgram       udp      wait      root      internal
#daytime   stream      tcp      nowait    root      internal
#daytime   dgram       udp      wait      root      internal
#chargen   stream      tcp      nowait    root      internal
#chargen   dgram       udp      wait      root      internal
#time      stream      tcp      nowait    root      internal
#time      dgram       udp      wait      root      internal
#
# ftp et telnet sont deux services très utilisés. Ils ne sont pas
spécialement sécurisés.
# telnet peut être remplacé par ssh qui est beaucoup plus sécurisé.
# ftp est une faille de sécurité importante remplacez le par sftp.
#
ftp        stream      tcp      nowait    root      /usr/sbin/tcpd
in.ftpd
telnet     stream      tcp      nowait    root      /usr/sbin/tcpd
in.telnetd
#
# Shell, login, exec, comsat et talk sont des protocoles BSD. Essayez de
ne pas les utiliser. Ils présentent des failles au niveau de la
sécurité.
#
shell      stream      tcp      nowait    root      /usr/sbin/tcpd
in.rshd
login      stream      tcp      nowait    root      /usr/sbin/tcpd
in.rlogind
#exec      stream      tcp      nowait    root      /usr/sbin/tcpd
in.rexecd
#comsat    dgram       udp      wait      root      /usr/sbin/tcpd
in.comsat
talk       dgram       udp      wait      nobody.tty /usr/sbin/tcpd
in.talkd
ntalk      dgram       udp      wait      nobody.tty /usr/sbin/tcpd
in.ntalkd
#dtalk     stream      tcp      wait      nobody.tty /usr/sbin/tcpd
in.dtalkd
#
# Pop3 et imap sont les serveurs de messagerie. A n'activer que si vous
les utilisez. Oubliez pop2
#
#pop-2     stream      tcp      nowait    root      /usr/sbin/tcpd
ipop2d
#pop-3     stream      tcp      nowait    root      /usr/sbin/tcpd
ipop3d
#imap      stream      tcp      nowait    root      /usr/sbin/tcpd
imapd
#
# Le service UUCP est un moyen d'envoyer des fichiers entre machines. Ce
service n'est pratiquement plus utilisé.
# Evitez de l'utiliser.

```

```
#
#uucp      stream      tcp      nowait      uucp      /usr/sbin/tcpd
/usr/lib/uucp/uucico
-l
#
# Tftp et bootp sont utilisés pour permettre aux machines clientes qui
# ne disposent pas de disque de booter, de recevoir une adresse IP, de
# charger le système.
# TFTP ne disposant pas de système d'authentification il est un énorme
# trou de sécurité.
#
# Vous devez absolument éviter de l'utiliser
#
# tftp      dgram      udp      wait      root      /usr/sbin/tcpd
in.tftpd
# bootps    dgram      udp      wait      root      /usr/sbin/tcpd
bootpd
#
# Finger, cfinger, systat and netstat ne sont pas dangereux en eux même,
# mais ils fournissent des informations sur les comptes
# utilisateurs et votre système. Il ne faut donc pas les utiliser.
#
# finger    stream      tcp      nowait      nobody    /usr/sbin/tcpd
in.fingerd
#cfinger    stream      tcp      nowait      root      /usr/sbin/tcpd
in.cfingerd
#systat     stream      tcp      nowait      guest     /usr/sbin/tcpd
/bin/ps -auwx
#netstat    stream      tcp      nowait      guest     /usr/sbin/tcpd
/bin/netstat -f inet

# Service d'authentification auth fournit des information sur
# l'utilisateur
auth stream tcp wait root /usr/sbin/in.identd in.identd -e -o
#
# End of inetd.conf
linuxconf stream tcp wait root /bin/linuxconf linuxconf --http
```

Le # devant une ligne rend la ligne inactive, donc le service non disponible.

Si vous ne devez pas utiliser un service, rendez le inactif. Certains services utilisent TCP-Wrappers, mais ne se trouvent pas dans ce fichier. Je pense entre autre à SSH (qui doit alors être compilé avec l'option --with-libwrap), et certaine version de portmap.

```
ftp      stream      tcp      nowait      root      /usr/sbin/tcpd
in.ftpd
```

ftp : nom du service, tel qu'il est déclaré dans /etc/services

stream : type de service de transport de données (il existe stream pour tcp, dgram pour udp, raw pour IP)

tcp : nom du protocole tel qu'il existe dans /etc/protocols

wait : état d'attente, si l'état est wait inetd doit attendre que le serveur ait restitué la socket avant de se remettre à l'écoute. On utilise wait plutôt avec les types dgram et raw. l'autre possibilité est nowait qui permet d'allouer dynamiquement des sockets à utiliser avec le type

stream.

root : Nom de l'utilisateur sous lequel le daemon tourne

/usr/sbin/tcpd in.ftpd Chemin d'accès au programme in.ftpd lancé par inetd (il est possible ici d'ajouter les options de démarrage du programme).

5 Les fichiers Hosts.allow et Hosts.deny :

Vous trouverez ces deux fichiers dans le répertoire /etc.

Le premier fichier lu est hosts.allow, puis hosts.deny.

Si une requête est autorisée dans le fichier hosts.allow alors elle est acceptée, quelque soit le contenu du fichier hosts.deny.

Si une requête ne satisfait aucune règle, que ce soit dans hosts.allow ou hosts.deny alors elle est autorisée. En un mot si vous ne mettez rien dans hosts.deny, alors vous n'avez rien fait.

Voici un petit exemple qui dans des cas simples est suffisant :

```
# hosts.allow
```

```
ALL : LOCAL
```

```
in.ftpd : 192.168.0.,10.194.168.0/255.255.255.0, 192.168.1.1
```

```
in.telnetd : .ac-creteil.fr
```

On autorise tout les ports depuis un accès local, et on autorise ftp pour les machines venant du réseau 192.168.0.0, ainsi que les machines du réseau 10.194.168.0 avec une autre notation et enfin la seule machine qui a pour adresse 192.168.1.1

et le fichier hosts.deny

```
#hosts.deny
```

```
ALL:ALL
```

Le fichier hosts.deny est simple à comprendre, il interdit tout par défaut.

Le fichier hosts.allow indique les services que je veux autoriser (Le nom du service doit être identique au nom qui se trouve dans inetd.conf).

la syntaxe est la suivante :

daemon [,daemon,] : client [,client ,] [: option : option]

Cette syntaxe est identique dans les deux fichiers, hosts.allow et hosts.deny.

6 Pour aller plus loin

On peut contrôler plus finement les accès à sa machine en contrôlant le fichier de log, en envoyant un message à la personne qui cherche à se connecter, on peut aussi se faire envoyer des messages en utilisant la commande mail. Voici un exemple un peu plus complet et complexe que le précédent :

```
# Fichier hosts.allow
```

```
ALL: LOCAL .intranet.moi EXCEPT sousdomaine.intranet.moi : ALLOW
```

```
in.ftpd : ALL : banners /root/messages.txt : spawn (echo " Accès au serveur ftp par l'adresse" %a "le " 'date') >> var/log/ftp.log &
```

sshd : 192.168.0.

in.telnetd : 10.94.243.1 EXCEPT PARANOID : spawn (/bin/mail -s "Alert le nom d hote et l adresse IP ne correspondent pas" root@%H)&

La ligne 1 indique que tous les ports sont ouverts pour la machine LOCAL et pour le domaine .intranet.moi sauf pour sousdomaine.intranet.moi

La ligne 2 autorise toutes les connexions sur le service ftp, mais envoie un message sur la machine qui se connecte, reste à placer le texte dans le fichier message.txt. spawn vous permet de faire appel à la commande echo qui envoie un message dans le fichier de log ftp.log avec l'adresse de la machine qui se connecte %a et la date.

La ligne 3 indique que seul le réseau 192.168.0.0 peut se connecter via ssh à la machine.

La ligne 4 autorise la connexion en telnet depuis la machine 10.94.243.1 uniquement si l'adresse IP de la machine et le nom d'hôte correspondent. On envoie alors un message en utilisant la commande mail.

Le fichier hosts.deny restant avec ALL : ALL

Les variables que vous pouvez utiliser sont :

%a	L'adresse IP du client
%A	L'adresse IP du serveur
%c	Informations disponibles sur l'utilisateur
%d	Le nom du daemon
%h	Le nom du client ou son adresse IP si on ne peut avoir le nom
%H	Le nom du serveur ou son adresse IP si on ne peut avoir le nom
%n	Idem mais en vérifiant le reverse DNS
%N	Idem pour le serveur
%p	Le pid du daemen
%s	Informations disponibles sur le serveur
%u	Nom de l'utilisateur
%%	Caractère %

7 Les utilitaires de tcp wrappers

tcpdchk -av : permet de voir la configuration de tcp wrappers

tcpdmatch in.ftpd localhost pour simuler une connexion sur in.ftpd

8 voir aussi

xinetd un équivalent de tcp_wrappers, mais en mieux.

TP 1 : Objectif : Protéger votre machine afin d'interdire les accès telnet et ftp depuis les autres machines

de la salle de formation sauf la machine de votre voisin. Bloquer pour tout le monde les services inutiles. Utiliser les utilitaires que vous connaissez afin de vérifier cela.

Faire ajouter l'adresse IP des clients non autorisés dans un fichier de log.

© [Philippe Chadeaux](#) - 10/10/2000 -



Syslog

1 Pour quoi faire

Syslogd est un daemon qui journalise les événements du système. Il faut avoir le daemon syslogd qui tourne sur votre machine pour que cela marche. Lorsque vous lancez syslog sur votre machine vous démarrez en fait le daemon syslogd et klogd, qui logue plus précisément les messages d'erreur du noyau. Vous avez donc en fait deux daemon qui tournent sur votre machine.

Tous les programmes tournant sur votre machine n'utilisent pas syslog. Par exemple Squid qui place ses logs dans access.log, mais aussi le serveur apache.

2 Le principe

Par défaut les fichiers de log se trouvent dans /var/log. Le fichier de configuration de syslog est dans /etc/syslog.conf. Par mesure de sécurité, il est d'usage de mettre le répertoire /var/log dans une partition propre (afin d'éviter qu'une saturation de ce répertoire n'entraîne un arrêt du système tout entier).

Les fichiers de log sont les suivants :

/var/log/messages est le fichier système qui récupère tout. On trouve aussi les messages des programmes qui utilisent syslog, à savoir par exemple named, sendmail.

/var/log/secure Contient les informations de connexions. Chaque login y est enregistré.

/var/log/maillog Contient un enregistrement du trafic de courrier entrant et sortant.

/var/log/spooler Contient les messages d'erreur des daemons uucp et innd (news).

3 L'installation

Par défaut il est installé avec la plupart des distributions, mais au cas où, peut-être qu'il faut l'installer :

syslogd-.....rpm utilisez la dernière version. Normalement il est lancé au démarrage de la machine. Si cela n'est pas le cas vous pouvez le lancer avec la commande
/etc/rc.d/init.d/syslog start.

4 Configuration

La configuration de syslog se fait dans le fichier /etc/syslog.conf. Pensez après toute modification à faire relire ce fichier de conf (killall -HUP syslogd).

Dans le fichier `syslog.conf` vous devez donc indiquer sur une ligne : le service, le niveau de gravité et le fichier vers lequel diriger les logs (cela peut être la console).

Les différentes catégories de service sont :

auth ou security	Messages de sécurité et d'authentification.
authpriv	La même chose que précédemment, mais logs plus privés
cron	Messages de crontab et de at
daemon	Messages systèmes générés par le daemon
ftp	Messages du serveur ftp
kern	Messages du noyau
lpr	Messages du serveur d'impression
mail	Messages du serveur de messagerie
news	Messages du serveur de news
syslog	Messages de syslog lui-même
user	Messages générés par le programme en cours d'un utilisateur
uucp	Messages UUCP

Puis la liste de sévérité, classée de la moins grave à la plus grave. Tous les messages de sévérité plus graves sont inclus, ainsi si vous choisissez sévérité `err`, vous avez aussi les messages `crit`, `alert`, et `emerg`.

7	debug	Messages de débogage
6	info	Messages d'information
5	notice	Messages un peu plus importants que les messages info
4	warning ou warn	Messages d'avertissement
3	err	Messages d'erreur
2	crit	Situation critique
1	alert	Situation critique nécessitant une intervention immédiate
0	emerg ou panic	Système inutilisable

6 Exemple

Log tous les messages du noyau vers la console.

En plus de les envoyer sur la console je les dirige vers le fichier `/var/log/kernel`

```
kern.*                /dev/console
kern.*                /var/log/kernel
```

Log tous les messages dans le fichier `messages` à partir du niveau `info`

```
# (il ne manque donc que debug par rapport au tableau précédent)
# Sauf les messages du mail, que l'on place dans un autre fichier et les messages authpriv

*.info;mail.none;authpriv.none      /var/log/messages

# Placer ici les messages que seul l'administrateur a le droit de voir.
# authpriv donne les connexions infructueuses, les connexions avec la commande su.
authpriv.*                          /var/log/secure

# Log tous les messages de mail et les place dans le fichier maillog..

mail.*                              /var/log/maillog

# Log tous les messages d'urgence rendant le système instable dans tous les fichiers.

*.emerg                             *

# Log les messages uucp et news dans un fichier spécial

uucp,news.crit                      /var/log/spooler

# Un truc que j'aime bien. Dirige tous les messages vers la console 12. On peut l'adapter.
# Avantage vous n'êtes pas obligé d'ouvrir une session pour avoir les messages à l'écran.
# Attention toutefois à la sécurité et confidentialité de cela.

*.*                                /dev/tty12
```

7 Remarques divers

- Si vous indiquez un fichier qui n'existe pas, syslog ne sera pas en mesure de le créer. Faites un touch /var/log/mon_fichier pour créer le fichier avant.
- N'hésitez pas à utiliser plusieurs fichiers pour bien isoler les messages importants des messages normaux.
- Pensez à utiliser une partition spéciale pour placer vos fichiers de log, afin d'éviter de voir votre système s'arrêter en raison d'une saturation du disque.
- Pensez à vérifier que vos fichiers ne deviennent pas trop volumineux. Faites les tourner (logrotate) régulièrement.
- Rien ne sert de créer des fichiers de log si vous ne les lisez jamais. Il existe des outils pour en extraire les messages importants, voire se faire envoyer les messages importants, à savoir autobuse, logcheck, swatch.

8 Tester votre configuration

Une fois votre fichier syslog.conf configuré, vous pouvez le tester en utilisant la commande logger. Logger est un utilitaire fourni avec syslog, qui vous permet d'envoyer des messages directement à syslog.

logger -p ftp.info "Message pour voir".

L'option -p permet d'indiquer le niveau de priorité. Par défaut user.notice

L'option -f permet d'indiquer un fichier.

L'option -t permet d'indiquer un tag **logger -t Test "Voici le message"**

9 Exporter vos logs sur une autre machine

Il est possible avec syslog d'envoyer ou de concentrer les logs sur une machine distante.

Par exemple si vous souhaitez avertir les personnes connectées via un terminal (root, toto, moi) sur votre serveur linux, sur les messages d'erreur du noyau, vous pouvez mettre :

```
kern.crit      root, toto, moi
```

L'autre possibilité, et de loin la plus intéressante, est de pouvoir centraliser tous les messages de vos serveurs linux sur une seule machine. En cas d'attaque d'une de vos machines, les logs se trouvant sur une autre machine, cela vous laisse une chance qu'ils soient intacts, et donc de pister l'incident. Pour mettre cela en place il vous faut ajouter dans

/etc/rc.d/init.d/syslog l'option -r derrière syslogd afin que celui-ci sache qu'il faut qu'il écoute sur le port 514 UDP. Sur la machine qui doit envoyer les messages, indiquer dans le fichier /etc/syslog.conf

```
kern.crit      @l'autre_machine.
```

Attention toutefois à ne pas permettre à la terre entière d'envoyer des messages sur cette machine.

TP 1 : Configurer syslog afin d'avoir les messages du noyau envoyés dans le fichier stage.

TP 2 : Ajouter une ligne afin de pouvoir voir les messages d'erreurs, à partir de la gravité "notice", arriver sur la console 12.

TP 3 : Si le serveur Squid tourne sur votre serveur pouvez vous faire apparaître les messages de connexion sur la console 11.

TP 3 : Choisir la machine stage1 afin de faire arriver l'ensemble des messages sur cette machine. Tester en utilisant la commande logger.



Crontab - at

1 Pour quoi faire

Crontab est un utilitaire bien utile et plutôt simple à mettre en oeuvre. Il permet de programmer des actions régulières sur votre machine. Par exemple est ce que tel process tourne toujours, est ce que ma ligne ADSL est toujours active, éventuellement faire des sauvegardes.

at permet quant à lui de lancer des actions à une heure donnée, un jour donné, mais sans répétition.

2 L'installation

Pas grand chose à dire sur l'installation de crontab sur votre machine, ni même de la commande at. Par défaut cela tourne déjà. Si ce n'est pas le cas installer les paquetages suivants :at-3.1.7-....rpm et crontabs-1.7-8mdk....rpm.

Vous devez voir le démon crond et atd tourner sur votre machine, si vous souhaitez les utiliser.

3 crontab

Un demon nommé cron lit le fichier (qui se trouve dans le répertoire /var/spool/cron) et exécute les commandes qui s'y trouvent.

Pour créer ce fichier taper **crontab -e** (crontab est une commande). Vous ouvrez alors vi et il vous reste à entrer votre ligne en respectant la syntaxe. Une fois sorti de vi (Escape puis :wq!), votre commande est mémorisée (vous obtenez le message crontab: installing new crontab). Pour visualiser toutes les crontab tapez **crontab -l**.

Lorsque vous créez une crontab, elle est créée pour l'utilisateur que vous êtes. Si vous souhaitez voir, créer, modifier ou détruire une crontab d'un autre utilisateur indiquer **crontab -u toto -l** (-l pour voir, -e pour créer ou modifier, -r pour détruire).

La syntaxe des entrées est la suivante :

<minute> <heure> <jour du mois> <mois> <jour de la semaine> <commande>
(avec un espace entre chacun)

minute : de 0 à 59

heure : de 0 à 23

jour du mois de : 1 à 31

mois de : 1 à 12

jour de la semaine : de 0 à 6, 0 étant le dimanche et ainsi de suite.

commande : peut comporter plusieurs commandes.

les mois et les jours peuvent aussi être donnés avec les abréviations anglaises :jan,feb,... et mon,tue,...

On sépare les jours, les mois par des , (*virgules*), par exemple pour lancer une action tous les 15 et 30 du mois tapez 15,30 à la place de jour du mois.

Le - signifie jusqu'à, ainsi 15-30 signifie du 15 au 30.

Le / permet de spécifier une répétition, */3 indique toutes les 3 minutes.

* peut être utilisée et indique tous les jours de semaine, tous les mois, toutes les heures.

3 Exemples

0 1 1 * * commande veut dire que vous n'exécutez que le premier jour du mois à 1 heure.

0 1 * * mon commande veut dire une fois par semaine le lundi à 1 heure.

0 1 1,15 * * commande veut dire tous les 1 et 15 du mois à 1 heure.

0 1 1-15 * * commande veut dire tous les 15 premiers jours du mois à 1 heure.

0 1 */5 * * commande veut dire tous les 5 jours à 1 heure.

***/3 * * * * commande** veut dire toutes les trois minutes.

La commande suivante efface tous les jours, les fichiers présents dans le répertoire /var/log depuis plus de 7 jours.

```
0 1 * * * find /var/log -atime 7 -exec rm -f {} \;
```

atime permet de trouver les fichiers non utilisés depuis 7 jours.

4 at

La commande at permet de lancer une commande un jour donné, à une heure donnée. Une fois que cette commande a été exécutée, elle n'existe plus. Pour utiliser la commande at taper at puis l'heure.

ainsi at 12:30 déclenchera la commande à 12 heures 30.

La syntaxe des entrées est la suivante :

at 12:30 11/30/00 déclenchera la commande le 30 novembre 2000 (le jour étant indiqué sous la forme mm/jj/aa.

at now + 1 hour déclenchera la commande dans 1 heure à partir de maintenant.

at 00:00 + 2 days pour exécuter la commande dans 2 jours à minuit.

Jusque là nous n'avons entré aucune commande à exécuter. Lorsque vous tapez la commande at 12:30, vous obtenez l'invite de la commande at
at 12:30

```
$ at 12:30
```

```
at>ping -c 1 192.168.0.1
```

```
at> ^D
```

```
$
```

vous avez alors le message suivant, qui vous indique que votre demande a été prise en compte, avec numéro d'ordre le 1.

```
job 1 at 2000-11-10 12:30
```

La commande va envoyer un ping sur la machine 192.168.0.1 à 12 heures 30 (j'avais pas d'autres idées sur le moment...!). On peut bien sûr faire exécuter un script. Puis at envoie par mail le résultat de cette commande.

Si vous souhaitez voir ce qui va se passer tapez at -c 1 (1 étant le numéro d'ordre).

Si vous ne connaissez plus toutes les commandes en attente tapez at -l (ou atq).

Pour supprimer une commande en attente atrm 1 (pour supprimer le job 1).

Attention un des grands risques lorsque l'on utilise at est de ne pas vérifier l'heure et le jour de l'exécution de la commande.

5 Contrôle

Dans le cas de at ou de crontab, il est possible de définir qui a le droit d'utiliser ces commandes. Pour cela il existe les fichiers /etc/cron.allow /etc/cron.deny et pour at /etc/at.allow et /etc/at.deny. Pour interdire par exemple l'utilisation de la commande at saisissez le nom des utilisateurs dans le fichiers at.deny (un par ligne).

6 Remarques

Il existe des interfaces graphiques pour créer, modifier, enlever des crontab, mais est-ce bien nécessaire ?

Si votre machine n'est pas allumée en permanence, il est alors possible que votre commande crontab ne puisse s'exécuter. Vous pouvez alors utiliser anacron (qui va vérifier les crontab non exécutées et les exécuter dès la remise en route de la machine).

TP 1 : Créer une crontab qui puisse vérifier toutes les deux minutes si squid tourne sur votre machine, sinon relancer squid, et envoyer un message à l'administrateur.

TP 2 : Créer une crontab qui sauvegarde et tar votre répertoire home, toutes les nuits. (on ne souhaite pas perdre les fichiers des nuits précédentes).

TP 3 : Créer une crontab qui surveille si la ligne adsl est toujours montée, si non la remonte.



Ipchains

1 Pour quoi faire

Ipchains est un outil permettant de filtrer les paquets. Il est donc possible avec Ipchains de contrôler quelles machines peuvent se connecter, sur quels ports à votre machine, mais aussi quelle machine de réseau peut aller sur tel serveur extérieur.

Ipchains est aussi utilisé pour faire du masquarading, c'est à dire permettre de partager une connexion à internet en utilisant une seule adresse IP, et donc en masquant les machines de votre réseau local.

Vous l'avez bien compris l'utilité d'ipchains est de pouvoir construire un firewall, et par les temps actuels avec l'arrivée de l'ADSL et du câble qui permettent d'avoir des connexions permanentes, il est souhaitable de mettre en place ce genre de choses.

Cet outil n'est pas le plus simple à mettre en oeuvre. Il demande de bien maîtriser TCP/IP. Il existe un certain nombre d'outils pour vous aider à utiliser ipchains, car comme nous allons le voir les règles ne sont pas simples à mettre en oeuvre. Je pense ici à l'interface graphique gfwc entre autre.

Ipchains remplace ipfwadm dans les versions actuelles de Linux et sera remplacé par netfilter à partir de la version 2.4 du noyau... donc prochainement.

Ce document n'a pas pour but de vous faire maîtriser toutes les possibilités d'ipchains, mais de vous permettre d'en comprendre le principe. Il existe un certain nombre de scripts que vous pouvez trouver sur internet qui font cela pour vous.

2 Le principe

Le principe est assez simple à comprendre, il est malheureusement moins simple à mettre en oeuvre.

Un paquet arrive sur votre machine (on parle de paquet entrant INPUT), vous devez alors choisir ce que vous en faites. Vous pouvez accepter (ACCEPT), vous pouvez rejeter (REJECT) ou le denier (DENY). La différence entre les deux derniers modes, est de prévenir l'envoyeur ou pas, que son paquet a été refusé (REJECT on prévient, pas avec DENY).

Une fois le paquet arrivé sur votre machine, il peut être forwardé (FORWARD) sur une autre carte réseau par exemple. Enfin arrivé sur cette deuxième carte réseau il peut être autorisé à sortir ou non.

Cette dernière notion est souvent plus compliquée à comprendre, toutefois imaginé que votre

machine autorise un type de paquet, un pirate arrive à prendre une partie du contrôle de votre machine, tant qu'il n'est pas en mesure de modifier les règles, il ne peut pas rebondir sur les autres machines de votre réseau.

Enfin dans le mode FORWARD on dispose d'une option supplémentaire qui est MASQ, ce qui permet de masquer les adresses IP des machines se trouvant dans votre réseau local. Cela permet entre autre de laisser croire à votre provider que vous disposez que d'une seule machine pouvant se connecter à l'internet.

3 L'installation

L'installation ne pose aucun problème, utilisez le RPM ipchains le plus récent. Installez vous aussi gcc qui va grandement vous aider dans cette épreuve. On trouve ces deux rpm dans toutes les distributions actuelles.

4 La syntaxe

La syntaxe d'ipchains est la suivante :

ipchains -A|I chaîne -i interface -p protocole -s adresse source port -d adresse destination port -j police -l

A chaîne	Ajoute une règle à la fin d'une chaîne. chaîne peut être de type INPUT, OUTPUT, FORWARD
I chaîne	Ajoute une règle au début d'une chaîne
-i interface	Pour indiquer l'interface eth0, eth1, ppp0, Io interface peut être remplacé par eth0, eth1, ppp0, Io.
-p protocole	Indiquer le protocole à savoir TCP, UDP, ICMP, ALL. Si rien n'est indiqué la règle s'applique à tous. On peut aussi utiliser à la place de protocole les noms et numéros se trouvant dans le fichier /etc/protocols
-s adresse source port	Indiquer l'adresse source du paquet. 0.0.0.0/0 pour n'importe quelle origine (ou any/0) 192.168.0.0/24 pour un réseau 192.168.0.0 avec un masque sur 24 bits Si aucun port n'est indiqué la règle s'applique à tous les ports. Si vous souhaitez indiquer une plage de ports placer un : entre le début et la fin.
-d adresse destination port	Indiquer l'adresse destination. Identique à adresse source
-j police	Indiquer ce que vous souhaitez faire de vos paquets police peut être de type ACCEPT, DENY, REJECT Si chaîne est de type forward vous disposez de MASQ
-l	Log dans /var/log/messages lorsque la règle est satisfaite

Remarques :

Vous pouvez indiquer le nom du port à la place du numéro, ainsi vous pouvez indiquer

www à la place de 80.

Vous pouvez indiquer des constantes afin de rendre la lecture de votre fichier plus simple. par exemple on peut définir la constante `academie="10.0.0.0/32"` et `ma_machine="192.168.0.1"`, puis utilisez là dans votre règle
ipchains -A input -p tcp -y -s \$academie -d \$ma_machine www -i eth0 -j ACCEPT
 (cette règle indique que j'accepte en entrée sur eth0 tout ce qui viens de 10.0.0.0/32 sur la machine 192.168.0.1 sur le port 80).

Vous pouvez utiliser `!` pour indiquer le contraire ainsi `! $academie` indique tout sauf les adresses 10.0.0.0

L'option `-y` permet de vérifier les paquets d'initialisation de la connexion.

Vous disposez encore d'un grand nombre d'options dont voici les principales :

-F	Vider les chaînes de toutes les règles
-X	Pour supprimer une chaîne
-L	Lister les règles
-D	Supprimer une règle
-N	Crée une nouvelle règle
-C	pour tester une règle
-P	Change la police d'une chaîne

Exemples :

ipchains -D input 1	Supprime la règle input 1. Faites un <code>-L</code> pour savoir son numéro On peut aussi indiquer la règle comme on l'a rentrée.
ipchains -L ipchains -L input	affiche toutes les règles affiche toutes les règles de la chaîne input
ipchains -N chainepourmoi	On crée ici une nouvelle règle
ipchains -X chainepourmoi	pour supprimer la chaîne que j'ai créé au dessus
ipchains -F input	vide toutes les règles de la chaîne input
ipchains -P input deny	Interdit tout par défaut

5 Avant de créer vos premières règles

Il faut savoir un minimum de choses avant de créer vos premières règles.

- Connaître les ports utilisés, quoique vous puissiez donner les ports sous la forme de nom comme ils sont indiqués dans le fichier `/etc/services` (on peut donc écrire telnet ou 23).
- Connaître les ports privilégiés, qui vont de 0 à 1023 et les ports sans privilèges, qui sont utilisés par le client pour établir la connexion à savoir 1024 à 65535.

- Savoir que lorsqu'une connexion s'établit entre un serveur et un client, un port est utilisé par le client dans la zone 1024:65535 afin d'obtenir la réponse, il faut donc tenir compte de cela afin de permettre la réponse du serveur vers le client. Utiliser netstat pour constater cela.

Ainsi je dois permettre à un client de venir sur ma machine avec un port sans privilège pour INPUT et

permettre à ma machine de recevoir sur un port avec privilège et
permettre au serveur de répondre (OUTPUT) sur un port avec privilège à une machine sur un port sans privilège.

- Autant il est possible de connaître le port utilisé par le serveur que vous souhaitez contacter autant il n'est pas possible de connaître le port utilisé par votre station pour établir la connexion. Il est seulement situé entre 1024 et 65535.
- Savoir que certains protocoles compliquent un peu la chose. Par exemple ftp utilise deux ports privilégiés, un pour les commandes (21) et l'établissement de la connexion, l'autre pour le transfert des données (20).
- Les serveurs de messagerie peuvent fonctionner de façon distincte. Vous utilisez un serveur SMTP pour envoyer, et pour recevoir vos messages, ou vous utilisez un client POP ou IMAP pour recevoir. Cela dépend de votre configuration.
- Que pour pouvoir utiliser des adresses de type www.ac-creteil.fr il vous faut pouvoir communiquer avec un serveur DNS. Là encore cela va dépendre si vous avez installé un serveur DNS sur votre réseau local. DNS interroge sur le port 53 en TCP et UDP. Mais il faut tenir compte du fait que la plupart des providers utilisent plusieurs serveurs de nom, en répartissant la charge entre plusieurs machines.
- Certain serveur nécessite l'ouverture du service AUTH (113), en output mais aussi en input.

Cela vous donne une petite idée de la raison pour laquelle cela est plus compliqué que prévu.

6 Créer des règles

Pour créer des règles afin de protéger votre machine, il faut respecter plusieurs choses.

- Créer vos règles dans un fichier et donner lui des droits d'exécution. Ne lui donner que les droits minimum afin que le fichier ne puisse pas être lu et modifié par tout le monde. Pour cela commencer par lui indiquer le shell utilisé.

```
#!/bin/sh
```

- Indiquer où se trouve l'exécutable sous la forme d'une variable (cela n'est pas obligatoire).
IPCHAINS=/sbin/ipchains

- Définissez vos constantes (idem vous pouvez choisir de ne pas utiliser des constantes). Cela est souvent pratique et simplifie votre travail. Il n'y a pas de norme en la matière, essayez seulement de pouvoir vous relire.

Vous les déclarez en début de fichier de la façon suivante :

```
localhost="127.0.0.1/32"
ip_internet="10.194.1.1"
ip_intranet="192.168.0.1"
net_intranet="192.168.0.0/24"
privports="0:1023"
unprivports="1024:65535"
interface_internet="eth0"   cela peut être ppp0 si vous avez une connexion ADSL.
interface_intranet="eth1"
interface_loopback="io"
Any="0.0.0.0/0"
```

- Effacer toutes les règles avant toute chose

Afin de partir d'une base propre et de savoir exactement ce que vous faites effacer toutes les règles qui pourraient exister sur votre machine.

\$IPCHAINS -F (j'utilise ici la variable pour appeler ipchains).

- Définir une politique par défaut

Le plus normal est de tout interdire par défaut et de n'autoriser que ce que vous souhaitez autoriser sur votre machine.

On deny tous les paquets entrant sans informer l'expéditeur, on "reject" tous les paquets en sortie et en forward afin d'être soi-même prévenu si une règle n'est pas correcte.

\$IPCHAINS -P input DENY

\$IPCHAINS -P output REJECT

\$IPCHAINS -P forward REJECT

Je vous conseille au départ d'accepter les output et forward, afin de ne pas compliquer la mise en place de votre protection.

Autoriser le trafic sur l'interface loopback

\$IPCHAINS -A input -i \$interface_loopback -j accept

\$IPCHAINS -A output -i \$interface_loopback -j accept

A partir de là, cela dépend de ce que vous souhaitez mettre en place, et de la configuration de votre machine, nombre de cartes réseau, type de connexion, réseau local ou machine isolée, se protéger contre les autres machines du réseau ou par rapport à l'internet.

Il n'est pas possible ici de traiter tous les cas possibles, ni de donner toutes les règles permettant de protéger votre machine ou réseau contre tous les types d'attaque. La mise en place d'une bonne sécurité repose sur la mise en place de plusieurs éléments.

7 Voici quelques règles types :

Autoriser telnet vers votre machine

```
$IPCAHINS -A input -p tcp -s $Any $privports -d $ip_internet telnet -i  
$interface_internet -j ACCEPT -l  
$IPCHAINS -A output -p tcp -s $ip_internet telnet !y -d $Any $unprivports -i  
$interface_internet -j ACCEPT
```

cette dernière ligne n'est nécessaire que si vous avez mis output REJECT, si par contre vous avez utilisé ACCEPT pour output, elle n'est pas nécessaire. Any ouvre la porte telnet à tout le monde, vous pouvez préciser les machines que vous souhaitez autoriser.

Le -l permet de loguer les paquets entrants.

Cette règle ne porte que sur la carte côté internet. Vous devez avoir la même côté intranet (côté votre réseau local) avec en plus la règle pour forward.

!-y permet de vérifier les bits indicateurs ACK, qui sont échangés entre le serveur et le client à chaque envoi de paquet TCP.

Autoriser telnet depuis votre machine (cas où vous n'avez pas mis accept pour output)

```
$IPCHAINS -A output -i $interface_internet -p tcp -s $ip_internet unprivports -d Any  
telnet -j ACCEPT  
$IPCHAINS -A input -i $interface_internet -p tcp ! -y -s $Any telnet -d $ip_internet  
unprivports -j ACCEPT
```

Autoriser les ping vers votre machine

```
$IPCHAINS -A input -p icmp -s $Any 8 -d $ip_internet -i $interface_internet -j  
ACCEPT  
$IPCHAINS -A output -p icmp -s $ip_internet 0 $Any -i $interface_internet -j  
ACCEPT
```

Cette dernière ligne n'est nécessaire que si vous avez mis output REJECT, si par contre vous avez utilisé ACCEPT pour output, elle n'est pas nécessaire.

Remarques :

- Le fait d'avoir deux cartes réseau ne change pas grand chose, sauf qu'il faut écrire un peu plus de règles. De plus il faut activer le forwarding entre vos deux cartes réseau, afin de permettre au client de votre réseau local de passer par votre firewall.

8 Masquerading

Le forwarding vous donne la possibilité de router les paquets du réseau interne vers le réseau externe (ou simplement entre deux réseaux). Vous pouvez autoriser une machine et une seule de votre réseau local à utiliser telnet sur une machine de votre réseau externe (ou l'inverse...enfin tout est possible).

Le masquerading est le fait de permettre aux machines de votre réseau interne de pouvoir sortir sur votre réseau externe en utilisant une seule adresse IP. Cette adresse officielle est mise à la place de l'adresse IP de votre machine cliente et re-remplacée au retour du paquet.

La différence entre Forwarding et Masquerading est qu'aucune de vos machines interne ne peut être atteinte par une machine de l'extérieur avec le masquerading.
 Pour pouvoir utiliser masquerading vous devez d'abord activer le forward. Vérifiez que vous avez bien dans /etc/sysconfig/network la ligne suivante : **forward_ipv4=yes**.
 Sinon pour l'activer tapez `echo "1" > /proc/sys/net/ipv4/ip_forward` (il faut redémarrer le réseau).

Il vous faut enfin ajouter la règle suivante, afin que les machines de votre réseau interne puissent en bénéficier.

\$IPCHAINS -A forward -i \$interface_internet -s \$net_intranet -j MASQ

- Si vous disposez d'une connexion ADSL pensez que l'interface est ppp0. Que cette interface n'est active que lorsque la connexion est établie, enfin que l'adresse IP obtenue peut être obtenue par dhcp.

9 Si vous êtes parano (j'en connais...!)

Pour essayer de protéger votre machine un peu mieux, il vous faut interdire un certain nombre de paquets auquel l'individu normal ne pense pas obligatoirement. Par exemple, interdire les paquets prétendant venir de vous, les paquets avec une adresse de classe A,B,C privée, les paquets qui arrivent avec comme adresse, l'adresse loopback, les paquets broadcast mal formés, une adresse multicast de classe D, une adresse réservée de la classe E, une adresse réservée par l'IANA.

Je ne peux pas ici les donner toutes. D'ailleurs je ne suis pas certain de tous les connaître. Mais il en existe encore un certain nombre.

Cela dépasse l'objet de cette formation.

Vous avez maintenant une petite idée de la difficulté à établir correctement une sécurité. Il existe des scripts pour faire cela. Je ne peux que vous conseiller de les utiliser.

TP 1 : Ce travail doit être réalisé par deux personnes. Lancer une connexion telnet sur la machine de votre voisin. Vérifier avec les outils dont vous disposez, les ports utilisés sur la machine locale et sur la machine distante.

Interdire tous les ports en input (par facilité autoriser les output et forward).

Autoriser la machine de votre voisin à se connecter sur le port telnet.

Pouvez vous faire du telnet sur la machine de votre voisin.

Autoriser les "pings" sur votre machine.

Pouvez vous pinguer depuis votre machine, que devez vous ajouter pour pouvoir le faire.

Si vous avez installé un proxy Squid sur votre machine, autoriser la machine de votre voisin à utiliser votre proxy (pensez au dns).

(On utilisera gfwc pour ce travail).



DHCP sous LINUX

La configuration du serveur DHCP consiste à configurer 2 fichiers:

- /etc/dhcpd.conf : ce fichier sert à la configuration même du serveur (plage d'adresses, paramètres distribués...)

- /etc/dhcpd.leases : ce fichier va servir à l'inscription des clients. Il peut ne pas se trouver dans ce répertoire, cela dépend de la version installée, on peut aussi le trouver dans /var/dhcpd. Chaque client DHCP, génère l'écriture d'un enregistrement dans ce fichier. Cela permet le suivi, les statistiques... de l'activité du serveur.

1. Installer le serveur dhcp

DHCP est le plus souvent installé par défaut sur votre machine, si cela n'est pas le cas installer la version rpm ou tar sur votre machine. Elle se trouve sur le CD de votre distribution préférée. Si vous utilisez une version RPM, vous avez alors un fichier /etc/rc.d/init.d/dhcpd pour démarrer votre serveur dhcp.

2. Le fichier dhcpd.conf

```
[root@mon_serveur_linux /etc]# more dhcpd.conf
# ici il s'agit du réseau 192.168.0.0
subnet 192.168.0.0 netmask 255.255.255.0 {
#La plage d'adresses disponibles pour les clients
range 192.168.0.10 192.168.0.200;
# Les clients auront cette adresse comme passerelle par défaut
option routers 192.168.0.254;
# Ici c'est le serveur de nom, le serveur privé, il faut aussi mettre le DNS donné par
votre provider. Pour Créteil #195.98.246.50 On peut en mettre plusieurs.
option domain-name-servers 192.168.0.1;
option domain-name-servers 195.98.246.50
# On donne le nom du domaine
option domain-name "ac-creteil.fr";
# Et l'adresse utilisée pour la diffusion
option broadcast-address 192.168.0.255;
#Le bail a une durée de 86400 s par défaut, soit 24 h
# On peut configurer les clients pour qu'ils puissent demander une durée de bail
spécifique
default-lease-time 86400;
```

```
#On le laisse avec un maximum de 7 jours
max-lease-time 604800;
}
```

Lorsque un client cherche à obtenir une adresse IP, il reçoit les informations suivantes :

Une adresse IP (la première de la liste dans votre plage d'adresse et ainsi de suite) :

192.168.0.10

La passerelle : 192.168.0.254

Le dns : 192.168.0.1

Le deuxième dns : 195.98.246.50

Le nom du domaine : ac-creteil.fr

L'adresse de broadcast : 192.168.0.255

La durée du bail ici une journée

La durée maximale d'un bail ici 7 jours

3. Création d'un fichier d'inscription /etc/dhcpd.leases

Ce fichier doit être créé, sans quoi le serveur DHCP ne pourra pas démarrer. Il suffit de créer un fichier vide (Les dernières versions le crée dans le répertoire /var/dhcpd/) . Pour cela taper la commande `echo > /etc/dhcpd.leases`. Le fichier est créé. Voici ce que l'on peut avoir dedans après l'inscription du premier client:

```
[root@mon_serveur_linux /etc]# more /etc/dhcpd.leases
lease 192.168.0.10 {
starts 1 1999/05/20 22:15:21;
ends 1 1999/05/20 22:15:38;
hardware ethernet 00:40:21:3c:f2:dd;
uid 01:00:40:21:3c:f2:dd;
client-hostname "Client1";
}
```

On distingue les informations suivantes (Début du bail, Fin du bail, adresse MAC du client, le nom d'hôte du client. Attention ce nom est différent du nom netbios utilisé sur les réseaux Microsoft.

Le serveur est configuré, il faut l'arrêter et le redémarrer.

- pour arrêter le service: `/etc/rc.d/init.d/dhcpd stop`

- pour démarrer le service : `/etc/rc.d/init.d/dhcpd start`

4. Fournir une adresse IP en fonction de l'adresse MAC du client

On peut ajouter dans le fichier `dhcpd.conf` une instruction propre à chaque client. Pour cela l'instruction pour identifier une station est la suivante :

```
host ma_station {
hardware ethernet 00:00:88:88:aa:aa;
fixed-address 192.168.0.2;
}
```

Dans l'exemple, host est le nom de l'instruction et ma_station le nom de votre client. Vous pouvez donner les informations que vous avez donné dans global, par exemple si vous souhaitez donner pour cette station un autre serveur DNS vous pouvez ajouter option domain-name-servers 193.10.0.1. Elles sont alors prioritaires par rapport aux options globales.

**TP 1 : Installer le serveur DHCP, puis le configurer sur votre machine.
Affecter une adresse statique à la machine de votre voisin.**

© [Philippe Chadeaux](#) - 10/10/2000 -



TCP/IP LINUX

Correction des TP

Fiche Les outils TCP/IP

TP1 : Changer l'adresse IP de votre machine

La commande est la suivante `ifconfig eth0 192.168.0.1 netmask 255.255.255.0`, puis afin d'avoir une modification définitive changer cette adresse dans le fichier `/etc/sysconfig/network-scripts/ifcfg-eth0`. Pensez à relancer le réseau.

TP2 : Recherche d'informations sur le domaine ac-creteil.fr.

Nous allons utiliser `nslookup`, même si certaines informations peuvent s'obtenir simplement à l'aide d'un ping.

Vérifier que `nslookup` est bien installé sur votre machine. Sinon installer le rpm `bind-util`.

Pour avoir l'adresse IP et le nom de la machine web de l'académie de Créteil, on peut faire **`nslookup www.ac-creteil.fr`**, on a alors le nom et l'adresse IP de la machine.

Nom des serveurs SMTP, pour cela on peut faire

`nslookup`

`>set type=MX`

`>ac-creteil.fr`

Essayez aussi la commande **`ls -t MX ac-creteil.fr`**

Adresse IP du proxy de l'académie de Créteil.

Si on essaye un ping `proxy.ac-creteil.fr`, on a deux réponses pour une même adresse. Les deux réponses ont comme adresse IP `192.168.74.40` et `41`, comme les deux adresses sont des adresses privées de classe C, il n'est pas possible de les pinguer.

Constatez avec la commande **`ls -t CNAME ac-creteil.fr`** que proxy est un CNAME.

TP 3 : Ecrire un petit script pour savoir quelle machine de votre réseau local monte la ligne pour aller sur internet

On utilise la commande `tcpdump` avec la syntaxe suivante :

`tcpdump -l -q -x dst 192.168.0.1`

`dst` correspond à votre routeur qui a comme adresse `192.168.0.1`. Pas besoin ici de mettre `src` pour avoir les machines de votre réseau.

`tcpdump -l -q -x dst 192.168.0.1 | grep \> > fichier_pour_enregistrer`

le grep est là pour enlever les lignes inutiles pour ce travail. Je veux seulement avoir l'adresse IP de la machine qui monte la ligne, avec l'heure et le protocole.

Voici une version plus sophistiquée (merci Jean)

```
#!/bin/bash
# Execution ./montee.ligne.sh ---> écrit montee.ligne.txt
# Cette solution n'est pas optimisée !
# et les fichiers temporaires generes
# servent a en suivre les etapes.
# on peut n'ecrire qu'une seule ligne avec des pipes :
# sed '/^[^0-9]/d' test | sed s/.[0-9][0-9][0-9][0-9][0-9]/"/ | sed '/^[a-z]/d' | sed s/'>.*'/' | sed s/\.([0-9]*)$/\1/ >
montee.ligne.txt

echo " Heure adresse IP Port" > montee.ligne.txt
echo "-----" >> montee.ligne.txt
sed '/^[^0-9]/d' test > filtre1
sed s/.[0-9][0-9][0-9][0-9][0-9]/"/ filtre1 > filtre2
sed '/^[a-z]/d' filtre2 > filtre3
sed s/'>.*'/' filtre3 > filtre4
sed s/\.([0-9]*)$/\1/ filtre4 >> montee.ligne.txt

cat montee.ligne.txt
```

Fiche Crontab

TP 1 : Vérifier si un programme tourne toujours sur votre machine.

```
#!/bin/sh
/sbin/pidof squid > /dev/null
if [ $? = 1 ]
then
    /etc/rc.d/init.d/squid stop
    /etc/rc.d/init.d/squid start
    echo "Redémarrage de squid" | mail -s "[Squid] Redémarrage de Squid" chadeaux@ac-creteil.fr
fi
Puis la crontab : */2 * * * * /bin/squid_surveille
Pensez à donner les droits à ce script.
```

TP 2 : Sauvegarder les répertoires personnels

```
#!/bin/bash
# fichier sauve_home.sh
date=$(date)
set -- $date
tar czvf /var/sauve/home.$3$2$6.tgz /home/*

# extrait de crontab de root (fichier /etc/spool/cron/root)
```

```
0 1 * * * /var/home/sauve_home.sh
```

TP 3 : Contrôler si votre ligne ADSL est toujours montée.

```
#!/bin/sh
if ! ping -c 1 195.98.246.50 > /dev/null 2>&1
then
    /usr/bin/killall pppd pppt
    sleep 40
    /usr/sbin/pppt 10.0.0.138
fi
```

Puis la crontab */3 * * * * /bin/reconnect.sh

On "ping" une machine de l'internet (195.98.246.50), si on n'a pas de réponse on kill pppd et pppt et on relance.

Pinguer la machine la plus proche de vous. 10.0.0.138 est l'adresse IP du modem ADSL par défaut. Il existe une méthode plus sympathique pour faire cela....mais pour l'exercice cela est correct.

Fiche TCP_Wrappers

TP 1 : Contrôler l'usage de telnet et de FTP

Vérifier dans le fichier /etc/inetd.conf que les lignes suivantes non pas de # devant.

```
ftp stream tcp nowait root /usr/sbin/tcpd in.ftpd
telnet stream tcp nowait root /usr/sbin/tcpd in.telnetd
```

Utiliser un cat inetd.conf | grep ftp

Placer un ALL:ALL dans le fichier hosts.deny afin de tout interdire sur votre machine.

Ajouter l'autorisation pour la machine de votre voisin de se connecter à votre machine en telnet.

Dans hosts.allow ajouter la ligne

```
in.ftpd : 192.168.0.2,127.0.0.1
```

Indiquer à la place de 192.168.0.2 l'adresse de la machine de votre voisin.

Idem pour telnet.

Demander à votre voisin de faire le test. Pensez éventuellement à relancer inetd par un killall -HUP inetd.

Ajouter dans le fichier ftp.log qui essaye de se connecter.

```
in.ftpd : ALL : spawn (echo " Accès au serveur ftp par l'adresse" %a "le " 'date') >> var/log/ftp.log &
```

Pensez à créer le fichier ftp.log avec un touch /var/log/ftp.log.

Fiche DHCP

TP 1 : Installer un serveur DHCP

Installer le rpm dhcp-3.0b1pl12...rpm si cela n'est pas déjà fait.

Créer le fichier dhcpd.leases avec la commande touch /etc/dhcpd.leases et laissez le vide. Il peut avoir été créé à l'installation du rpm dans le répertoire /var/dhcpd.

Créer le fichier /etc/dhcp.conf, il n'est pas installé avec le rpm.

```
option domain-name "ac-creteil.fr";  
option domain-name-servers 195.98.246.50;
```

```
subnet 192.168.0.0 netmask 255.255.255.0 {  
    range 192.168.0.10 192.168.0.200;  
    option routers 192.168.0.254;  
    option broadcast-address 192.168.0.255;  
    default-lease-time 86400;  
    max-lease-time 604800;  
}
```

Tester, puis essayer d'imposer une adresse IP à votre machine de tests.

```
host ma_station {  
    hardware ethernet 00:10:A4:98:EC:8E;  
    fixed-address 10.100.1.100;  
}
```

Remplacer ici l'adresse hardware par celle de votre machine.

Fiche Syslog

TP 1 : Envoyer les messages du noyau dans le fichier stage

Très simple. Ajouter la ligne kern.* /var/log/stage dans le fichier syslog.conf, relancer le daemon avec un killall -HUP syslogd.

Pensez à créer le fichier /var/log/stage avec un touch /var/log/stage.

TP 2 : Messages sur la console 12

Ajouter la ligne *.notice /dev/tty12 dans le fichier syslog.conf

TP 3 : Messages du serveur squid

Squid (comme apache) n'utilise pas syslog. Il place les logs d'accès dans le fichier /usr/local/squid/logs/access.log (cela est variable en fonction de ce que vous avez indiqué au moment de le compiler).

Ajouter une ligne comme tail -f /usr/local/squid/squid/access.log > /dev/tty11 & dans votre fichier /etc/rc.d/rc.local

TP 4 : Envoyer vos messages sur la machine stage1

Il faut commencer par modifier le fichier /etc/rc.d/init.d/syslog en ajoutant -r sur la ligne **daemon syslogd -r -m 0**. Relancer alors syslog, /etc/rc.d/init.d/syslog stop, puis start.

On constate avec la commande netstat que l'on a bien une écoute sur le port UDP 514.

Donc il reste à configurer la machine qui doit envoyer les messages.

Dans le fichier /etc/syslog.conf on ajoute la ligne ***.* @aleu** aleu est le nom de ma machine qui doit recevoir les logs de toutes les machines.

Envoyer un message **logger -t Test "Voila un texte pour tester"**.

Fiche Ipchains

TP 1 : Contrôler certains accès sur ou depuis votre machine

```
#!/bin/sh

IPCHAINS=/sbin/ipchains

localnet="192.9.200.0/24"
firewallhost="192.9.200.2/32"
Any="0.0.0.0/0"
localhost_localdomain="127.0.0.1/32"
dell_pc="10.100.1.19/32"

# Vide les règles existantes
$IPCHAINS -F

# Par défaut
$IPCHAINS -P input DENY
$IPCHAINS -P forward ACCEPT
$IPCHAINS -P output ACCEPT

# input rules

# Pour autoriser la machine de mon voisin à se connecter en telnet sur ma machine.
$IPCHAINS -A input -p tcp -s $10.100.1.5/32 1024:65535 -d $dell_pc telnet -i eth0 -j ACCEPT -l

# Pour m autoriser à faire du telnet sur les machines estérieures
$IPCHAINS -A input -p tcp -s $Any telnet -d $dell_pc 1024:65535 ! -y -i eth0 -j ACCEPT

# Pour autoriser d'autres machines à pinguer ma machine
$IPCHAINS -A input -p icmp -s $Any 8 -d $dell_pc -i eth0 -j ACCEPT

# Pour m'autoriser à pinguer toutes les machines extérieures
$IPCHAINS -A input -p icmp -s $Any 0 -d $dell_pc -i eth0 -j ACCEPT

# Pour autoriser dns en UDP
$IPCHAINS -A input -p udp -s $dnsserveur1 53 $dell_pc 1024:65535 -i eth0 -j ACCEPT

# Pour autoriser dns en TCP
$IPCHAINS -A input -p tcp -s $dnsserveur1 53 $dell_pc 1024:65535 ! -y -i eth0 -j ACCEPT

# Pour autoriser les machines à utiliser mon serveur Squid
$IPCHAINS -A input -p tcp -s Any 1024:65535 $dell_pc 8080 -i eth0 -j ACCEPT

# Pour autoriser mon serveur Squid à aller chercher sur le web
$IPCHAINS -A input -p tcp -s $dell_pc 1024:65535 $Any 80 ! -y -i eth0 -j ACCEPT

# forward rules
```

output rules

© [Philippe Chadeaux](#) - 10/11/2000 -



DNS sous LINUX

Bind

Un serveur DNS est un serveur qui transforme l'adresse de type <http://www.ac-creteil.fr> en adresse IP, qui sont les seules adresses valides sur internet. Lorsque vous souhaitez naviguer sur internet, vous devez obligatoirement avoir configuré le DNS de votre machine, sinon il vous sera impossible d'atteindre le moindre site web, à moins de connaître son adresse IP, ce qui est assez difficile à retenir. Le serveur DNS a une autre fonctionnalité qui est d'indiquer le serveur SMTP (serveur de messagerie) qui est autorisé à recevoir les messages pour votre domaine, ainsi lorsque vous envoyez un message à toto@ac-creteil.fr vous n'indiquez pas le serveur de l'académie de Créteil, qui a la charge de remettre ce message au bon serveur.

Le serveur DNS Bind que nous allons installer est le serveur le plus utilisé sur internet.

1. Objectif

La raison pour laquelle on peut avoir besoin d'un serveur DNS dans un établissement scolaire peut être double.

La première disposer d'un serveur cache DNS, afin d'accélérer les requêtes.

La seconde, qui n'est pas incompatible avec la première, est de simplifier l'adressage des machines internes à votre établissement.

Ainsi je veux que les élèves ne soient pas obligés de taper l'adresse IP de la machine web (intranet), mais puissent y arriver avec www.mon_lycee.fr, il faut alors que ce serveur puisse "forwarder" les requêtes vers un serveur dns officiel.

Il est clair qu'ici mon_lycee.fr est un sous domaine complètement inventé, il n'a pas de valeur légale sur internet. Ainsi je vous conseille vivement de ne pas utiliser un domaine existant.

Un domaine n'existe au sens légal que s'il a été déposé officiellement auprès d'un organisme autorisé (Voir le nic france pour plus d'informations). Cela serait la troisième possibilité de configuration d'un serveur DNS, vouloir installer un serveur public (officiel). Cette solution n'est pas envisageable pour un établissement scolaire.

Donc pour revenir à ce que l'on souhaite faire, on va se créer un domaine rien que pour nous, avec comme nom mon_lycee.fr et comme première machine [mon_serveur](http://mon_serveur.fr).

2. Installer le serveur bind

Pour installer bind, il vous suffit d'installer le rpm bind-8....rpm (je n'indique pas ici de version, utilisez de préférence la dernière), et rpm -i bind-8.....rpm et le paquet caching-nameserver (ce paquet n'est pas nécessaire mais il vous installe les fichiers named.conf et /var/named/named.ca et /var/named/named.local, il installe aussi named.boot qui n'est plus utilisé dans la version 8 de bind) il permet de configurer un cache dns (on peut bien sûr compiler les sources). En passant installer aussi le paquet bind-util nous l'utiliserons pour tester la configuration.

Vous obtenez alors les fichiers suivants :

/etc/named.conf Contient les paramètres généraux.
/var/named/named.ca Indique les serveurs dns racines.
/var/named/named.local résolution locale des adresses loopback

Il vous faut en fonction de ce que vous voulez faire créer les fichiers suivants :

/var/named/mon_lycee.fr fichier qui fait correspondre le nom de machine et son adresse IP
/var/named/db.192.168.0 fichier de zone inverse qui fait correspondre l'adresse IP avec le nom de machine.

3. Configurer Bind.

Il faut pour cela configurer les différents fichiers que nous venons de voir.
On cherche ici à configurer un domaine mon_lycee.fr avec comme adresse de réseau 192.168.0.0.

named.conf

```

;
;Fichier d'amorçage du serveur primaire pour
mon_lycee.fr
;
options {
    directory "/var/named";
    forward first;
    forwarders {
        195.98.246.50
    };
query-source address * port 53;
```

options définit les options du serveur dans son ensemble. On peut configurer plus finement en plaçant les options dans les zones (si vous gérez plusieurs domaines ou des sous domaines par exemple).

zone définit les options s'appliquant à des zones particulières.
La zone 0.0.127.in-addr.arpa crée une zone pour le réseau loopback.
La zone . indique l'emplacement du root du serveur du domaine internet. Un forward only ne nécessite pas de zone .
La zone mon_lycee.fr est la zone que vous souhaitez créer et qui a comme fichier mon_lycee.fr.

logging permet de configurer les logs de named. On peut les envoyer vers deux canaux syslog ou un fichier ou null.

```

allow-query {
    127/8;
    ! 192.168.1.10;
    192.168.1/24;
};

allow-transfer { ! *; };
allow-update { ! *; };

listen-on port 53 { *; };

};

logging {
    category statistics { null;
    };
    category security {
    default_syslog;
    default_debug;
    };
    category default {
    null;
    };
};

zone "." {
    type hint;
    file "named.ca";
};

zone "0.0.127.in-addr.arpa" {
    type master;
    file "named.local";
};

zone "mon_lycee.fr" in {
    notify no;
    type master;
    file "mon_lycee.fr";
};

zone "0.168.192.in-addr.arpa" in {
    notify no;
    type master;
    file "db.192.168.0";
};

```

Il existe plusieurs types de "**category**" (statistics, security, default,...). Vous pouvez paramétrer très finement cela.

directory indique le répertoire où se trouvent les fichiers. Vous pouvez à la place indiquer le chemin complet.

forward peut avoir plusieurs options (first, only) first redirige les requêtes aux serveurs se trouvant dans la liste forwarders, si les hôtes ne répondent pas, le serveur tentera de répondre. only redirige sans réponse aux serveurs se trouvant dans la liste forwarders

forwarders indique les serveurs vers lesquelles les requêtes sont envoyées. 195.98.246.50 est le dns de l'académie de Créteil.

query-source indique que le port 53 est le port d'échange (source et destination) entre les serveurs DNS. Très utile lorsqu'il y a un firewall.

allow-query contient une liste des adresses dont le serveur acceptera ou refusera les requêtes. L'ordre compte, le premier l'emporte. 127/8 autorise localhost, j'interdis la machine 192.168.1.10 et autorise les autres (un exemple seulement).

allow-transfer interdit les transferts de requête de zone. Par défaut cela est autorisé de partout.

allow-update refuse les instructions de mises à jour de la base de données de zone. Par défaut les mises à jour sont refusées.

listen-on port 53 indique le port en écoute pour les clients et les interfaces. Indiquer * pour écouter sur toutes les interfaces, ou l'adresse IP de la carte.

category statistics génère un rapport périodique d'activité.

category security requêtes acceptées/refusées. **category default** default est équivalent à toutes catégories.

type déclare le type d'entrée, il en existe de

plusieurs types (master, hint)

master déclare ce serveur comme étant primaire.
Si vous créez un serveur secondaire indiquez slave.

hint déclare que cette entrée n'est qu'un endroit où débiter les recherches.

notify no pour ne pas informer les autres serveurs s'il y a des changements dans la zone.

Named.ca

Je ne donne pas d'exemple ici. Vous n'avez pas à modifier ce fichier. Il contient les adresses des serveurs root.

named.local

```
@    IN    SOA mon_serveur.mon_lycee.fr.
postmaster.mon_serveur.monlycee.fr.(
        2000101500 ; numéro de série
        28800 ; rafraîchissement toutes les 8
heures
        14400 ; nouvel essai toutes les 4 heures
        604800 ; expiration dans 7 jours
        86400 ) ; temps de vie minimal 24
heures
NS    mon_serveur.mon_lycee.fr.
1     PTR localhost.
```

Normalement vous n'avez pas à changer les valeurs qui sont dans ce fichier.

La première partie est identique dans les trois fichiers, si vous devez faire une modification sur un fichier vous devez modifier le numéro de série afin de faire connaître cette modification aux autres serveurs dns.

20001015 correspond au 15 oct 2000 changer cela lorsque vous faites une modification. Si vous devez faire plusieurs modifications dans la même journée incrémenté le 00.

mon_lycee.fr

```
@    IN    SOA mon_serveur.mon_lycee.fr.
postmaster.mon_serveur.monlycee.fr.(
        2000101500 ; numéro de série
        28800 ; rafraîchissement toutes les 8
heures
        14400 ; nouvel essai toutes les 4 heures
        604800 ; expiration dans 7 jours
        86400 ) ; temps de vie minimal 24
heures
```

Vous indiquez dans ce fichier, les machines que vous souhaitez pouvoir appeler par leur nom (équivalent au fichier host enregistrement de type A).

; serveur de nom

```
IN NS mon_serveur.mon_lycee.fr.
```

;adresses IP des machines

```
localhost      IN A    127.0.0.1
mon_serveur    IN A    192.168.1.1
mon_serveur_web IN A    192.168.1.2
```

Indiquez aussi le serveur SMTP de votre domaine (enregistrement de type MX).

Les CNAME (alias) permettent de définir les alias sur des machines. Ainsi lorsque vous tapez www.ac-creteil.fr www est un alias sur la machine web du rectorat, qui possède en fait un autre nom.

<pre> ;Alias www IN CNAME mon_serveur_web ftp IN CNAME mon_serveur_web pop IN CNAME mon_serveur ; Serveur smtp mon_serveur_smtp IN A 192.168.1.3 IN MX 10 mon_serveur_smtp.mon_lycee.fr. </pre>	l'avantage étant de pouvoir changer de machine sans être obligé de faire de grosses modifications. N'hésitez donc pas à utiliser les alias. Ce fichier est celui que vous allez modifier le plus, pensez donc à changer le numéro de série.
db.192.168.0	
<pre> @ IN SOA mon_serveur.mon_lycee.fr. postmaster.mon_serveur.monlycee.fr.(2000101500 ; numéro de série 28800 ; rafraîchissement toutes les 8 heures 14400 ; nouvel essai toutes les 4 heures 604800 ; expiration dans 7 jours 86400) ; temps de vie minimal 24 heures ; serveur de nom IN NS mon_serveur.mon_lycee.fr. ; adresses IP inverses 1 IN PTR mon_serveur.mon_lycee.fr. 2 IN PTR mon_serveur_web.mon_lycee.fr. 3 IN PTR mon_serveur_smtp.mon_lycee.fr. </pre>	Fichier des reverses. Une entrée de type A dans ce fichier doit avoir une correspondance dans ce fichier, enfin normalement. Le 1, 2, 3 correspondent à respectivement l'adresse 192.168.1.1, et ainsi de suite.

4 Remarques :

- Pensez à toujours mettre un point à la fin des noms de machine + domaine.
- Vous n'aurez certainement pas le besoin d'un serveur SMTP, il est là pour l'exemple. Si vous deviez en ajouter un deuxième indiquer un poids supérieur (IN MX 15 mon_autre_serveur_smtp). Si vous souhaitez en faire votre SMTP principal indiquez un poids inférieur.
- Les numéros de série peuvent être différents d'un fichier à un autre. Vous n'avez qu'à modifier celui du fichier que vous modifiez.
- Postmaster doit être un compte existant sur votre machine. ce qui est normalement le cas. Il recevra tout le courrier concernant ce domaine.
- Un serveur DNS n'est pas simple à mettre en oeuvre. Il faut entre autre éviter de monter la ligne à chaque fois qu'une requête est envoyée au serveur web local.

Je n'ai pas ici donné toutes les options disponibles, mais toutefois , il peut être utile d'enregistrer certains logs de votre dns.

Voir les paramétrages du fichiers named.conf.

Attention à ne pas enregistrer toutes les requêtes et entre autres les requêtes qui aboutissent, car vous auriez alors des fichiers énormes.

Pensez à configurer votre fichier /etc/resolv.conf qui doit contenir les lignes suivantes :

domain mon_lycee.fr

nameserver 127.0.0.1 (autant utiliser le dns que vous venez de configurer, vous pouvez éventuellement en indiquer un autre).

5 Tester son serveur DNS

Une fois que votre installation est terminée, vous devez la tester.

Pensez avant tout à lancer le daemon named, sinon vous risquez de ne rien voir :-).

Pour cela, vous avez installé bind-util. Vous pouvez utiliser alors l'utilitaire [nslookup](#).

6 Configurer les clients

Je ne vais pas ici vous apprendre à configurer le dns de vos clients, par contre pensez à utiliser le dns, que vous venez de mettre en oeuvre, et donc de le placer en première position de vos dns.

TP 1 : Installer le serveur bind sur votre machine.

Configurer bind de façon à disposer d'un dns local capable de forwarder les requêtes officielles.



Squid

1 Pour quoi faire

Squid est un proxy cache sous linux. De ce fait il permet d'accélérer vos connexions à l'internet en plaçant en cache les sites les plus visités. Ainsi dans des établissements scolaires cela permet d'améliorer les connexions. On peut aussi effectuer des contrôles de sites, même si cela à mon avis reste du domaine de l'impossible. Enfin il est possible de partager une connexion à internet à l'aide Squid, mais Squid n'est pas un proxy POP, SMTP, NNTP (comme Sambar par exemple).

Avant de lire cette doc, qui est loin d'être complète, je vous conseille vivement de consulter la doc de Squid en anglais, mais fort bien faite et très complète. Voir aussi absolument le [site du cru](#) pour des conseils de configuration de votre machine. Il faut lire aussi les fichiers release1.0.txt et release1.1.txt qui se trouvent dans le répertoire /usr/doc/squidxxxxxx/.

2 Où trouver SQUID

Squid est disponible sur le cd de toutes les distributions de Linux. On peut aussi le télécharger sur le site de [Squid](#). On le trouve sous forme de fichiers tar ou rpm.

3 Avant d'installer SQUID

Peut être même avant d'installer votre machine linux, vous devez vous poser la question de quel type de machine, comment partitionner mon disque. Un disque SCSI est préférable à un IDE, un cache utilise beaucoup le disque il faut donc que celui-ci soit le plus rapide possible. Il est utile de disposer aussi de mémoire, cela ne peut qu'améliorer les performances de votre cache.

Pour ce qui est des partitions pensez à faire une partition pour le cache, éventuellement une pour placer les logs (ils sont nombreux avec squid), pour le reste à vous de voir.

Il me semble aussi qu'une deuxième carte réseau peut être utile dans ce type de cas.

Pensez aussi à sécuriser votre machine, un cache ne doit pas pouvoir être utilisable par la terre entière mais uniquement par des machines authentifiées et/ou autorisées.

4 Installer SQUID

Si vous utilisez les packages rpm il n'y a aucun problème à installer squid.

`rpm -ivh squid*.rpm` et le tour est joué.

Il place alors les fichiers de log dans /var/log/squid, le fichier de configuration dans /etc/squid/squid.conf. Ce fichier est le seul fichier de configuration de squid. Il faut donc ouvrir ce fichier pour effectuer les paramétrages correspondant à votre situation.

Si vous utilisez tar, vous devez faire `tar xvfz squid-2.2*.tar.gz`, puis lancer la compilation (lire le fichier Install avant ou utiliser `./configure --help` pour avoir les options de compilation).

`./configure` Squid sera installé par défaut dans le répertoire /usr/local/squid. Pour l'installer dans un autre répertoire compilez squid avec `--prefix=/la_ou_je_veux`

Si vous souhaitez avoir les messages d'erreurs en français ajouter `--enable-err-language=French` (On peut aussi faire cela dans le fichier squid.conf).

Si vous souhaitez utiliser un grand nombre d'ACL utilisez `--enable-gnuregex`, mais il est peut être souhaitable d'utiliser un produit comme squidguard (voir plus loin). Puis faire :

`make all`

`make install`

5 Démarrage de Squid

Avant tout démarrage de Squid il est nécessaire de le configurer. Pour cela il n'existe qu'un seul fichier squid.conf que vous trouverez dans le répertoire /etc/squid (si vous avez utilisé la version rpm) ou dans /usr/local/squid/etc (si vous avez compilé squid à partir d'un .tar).

Lors du premier démarrage de squid, il est nécessaire de créer les répertoires de swap avec la commande squid -z (ou quand vous modifiez la configuration du cache_dir).

Après il ne reste plus qu'à démarrer Squid avec la commande : /etc/rc.d/init.d/squid start (Attention le fichier squid.ini qui se trouve dans le répertoire /etc/rc.d/init.d/ n'est pas mis en place si vous avez vous même compilé squid).

On peut vérifier que tout est Ok en allant voir le fichier cache.log qui se trouve sur la Red Hat dans le répertoire /var/log/squid/

Par défaut Squid ne démarre pas automatiquement au démarrage de votre machine. Il faut donc le placer dans le bon niveau de démarrage. Voir un fichier de lancement [ici](#).

Il faut absolument éviter de démarrer Squid avec le compte root, pour cela utilisez plutôt le compte nobody ainsi dans le fichier squid.conf indiquer cache_effective_user nobody.

Il faut alors penser à donner les droits nécessaires à nobody pour que cela marche. Si vous avez vous même compilé Squid pensez à rendre nobody propriétaire de /usr/local/squid avec un truc comme chown -R nobody.nobody /usr/local/squid.

Pensez aussi à donner les droits qu'il faut dans le répertoire de cache qui comme il se doit se trouve sur une partition propre (même type de commande). Pensez éventuellement à modifier les droits.

Pour démarrer Squid on peut aussi utiliser RunCache, qui possède la particularité de relancer Squid lorsque celui-ci s'arrête. Pour cela lancer le avec une commande comme su - nobody -c /usr/local/squid/bin/RunCache& (le & est ici important il permet de faire tourner votre script en tâche de fond et donc de reprendre la main). Placer alors cette ligne dans rc.local.

Si Squid a bien démarré vous devez avoir en tapant ps ax |grep squid une réponse vous indiquant le numéro du process.

Si par la suite vous devez modifier votre fichier de configuration vous pouvez forcer la relecture de ce fichier avec un kill -HUP xxx (xxx étant le numéro de process).

6 Options de Squid

On peut aussi démarrer squid en lui passant des commandes sur la ligne de commande.

Différents paramètres peuvent être passés sur la ligne de commande. Les options passées de cette façon écrasent les paramètres du fichier squid.conf.

-h : Pour obtenir les options possibles

-a : Pour indiquer un port particulier

-f : pour utiliser un autre fichier de conf à la place de squid.conf

-i : désactive le cache IP

-u : spécifie un port pour les requêtes ICP.

-v : pour indiquer la version de Squid

-z : Pour effacer le contenu du cache sur le disque ou pour créer le fichier de swap.

-k : Pour envoyer des instructions à Squid pendant son fonctionnement. Il faut faire suivre -k d'une instruction (rotate|reconfigure|shutdown|interrupt|kill|debug|check-).

-D pour démarrer squid lorsque vous n'êtes pas connecté en permanence à internet (évite de vérifier si le serveur DNS répond).

7 Contrôler si Squid tourne

On peut pour cela utiliser cron et vérifier à l'aide d'un script si Squid est toujours en activité.

Voici un script possible pour faire cela.

```
#!/bin/sh
```

```
/sbin/pidof squid > /dev/null
```

```
if [ $? = 1 ]
```

```
then
```

```

/etc/rc.d/init.d/squid stop
/etc/rc.d/init.d/squid start
echo "redemarrage de Squid"

```

fi

Il existe bien d'autres possibilités et les puristes vous diront "et si cron est aussi dans les choux". On trouve alors sur Internet des outils pour surveiller des machines depuis d'autres machines. On peut aussi ajouter la ligne suivante afin d'être prévenu d'un redémarrage de Squid
 echo "Redémarrage de Squid" | mail -s "[Squid] Redémarrage de Squid " chadeaux@ac-creteil.fr.
 Vous pouvez aussi utiliser RunCache (voir plus haut), qui essaye de relancer le service.

8 Configuration de Squid

Toute la configuration de Squid se trouve dans le fichier Squid.conf que l'on trouve dans /etc/squid/ ou /usr/local/squid/etc.

A partir de la version 2 voici un fichier [squid.conf](#) avec des commentaires qu'il faut adapter à votre configuration.

La plupart des options par défaut du fichier Squid.conf ne sont pas à changer (vous pouvez alors laisser le # pour conserver les options par défaut).

http_port: le port que vous souhaitez utiliser. Le plus fréquent est 8080. Il faut donc changer cette valeur car par défaut Squid utilise 3128.

icp_port: Conserver le port 3130. Ceci vous permet de communiquer avec des proxy-cache parents ou voisins.

cache_mem : correspond au cache mémoire, la valeur dépend de votre système. Par défaut squid utilise 8 Mo. Cette taille doit être la plus grande possible afin d'améliorer les performances (Considérez 1/3 de la mémoire que vous réservez à Squid). Il faut avec cache_mem régler cache_mem_low et cache_mem_high qui sont les valeurs limites de remplissage du cache mémoire. Par défaut les valeurs sont 75 % et 90 %. Lorsque la valeur de 90 % est atteinte le cache mémoire se vide jusqu'à 75 %. Les valeurs par défauts sont bien dans la plupart des cas.

cache_swap : correspond à la taille de votre cache disque. Si la taille du disque le permet, et en fonction de la taille de votre établissement (nombre de client qui utilise le cache), mais aussi de la durée de rafraîchissement de votre cache et du débit de votre ligne, vous devez mettre la valeur qui vous semble correspondre à votre situation.

cache_peer : Indiquer ici les proxy parents du rectorat. Attention vous ne pouvez y accéder que si vous êtes connecté via le rectorat ou Oléane. cache_peer proxy.ac-creteil.fr parent 8080 3130 no-query default. Si vous utilisez un provider privée ne rien mettre ici.

acl QUERY urlpath_regex cgi-bin !? \.cgi \.pl \.php3 \.asp : Type de page à ne pas garder dans le cache afin de pas avoir les données d'un formulaire par exemple.

maximum_object_size : taille maximale de l'objet qui sera sauvegardé sur le disque. On peut garder la valeur par défaut.

cache_dir : Vous indiquez ici le volume de votre cache. Si vous avez plusieurs disques utilisez plusieurs fois cette ligne.

cache_dir ufs /cache1 100 16 256 (cache de 100 Mb)

cache_dir ufs /cache2 200 16 256 (cache de 200 Mb)

Placer de préférence le cache sur une partition propre.

cache_access_log ; cache_log ; cache_store_log : Indique l'endroit où se trouve les logs. Si vous ne souhaitez pas avoir de log (par exemple des objets cache_store_log) indiquer cache_store_log none.

debug_options ALL,1 : niveau de debug. Indiquer 9 pour avoir toutes les logs à la place de 1. Attention cela donne de gros fichiers.

ftp_user : A utiliser pour indiquer au serveur ftp qui demande une authentification.

ftp_user squid@ac-creteil.fr par exemple.

dns_children : Par défaut le nombre de requêtes dns est de 5. Il peut être nécessaire d'augmenter ce nombre afin que Squid ne se trouve pas bloqué. Attention de ne pas trop l'augmenter cela pouvant poser des problèmes à votre machine (indiquer 10 ou 15).

request_size : Taille maximale des requêtes. Conserver le défaut, concerne les requêtes de type GET,

POST..

refresh_pattern : Permet de configurer la durée de mise à jour du cache. Utiliser -i pour ne pas tenir compte des minuscules/majuscules. (voir le fichier squid.conf). Les valeurs Min et Max sont indiquées en minutes.

visible_hostname : indiquer ici le nom de votre serveur proxy.ac-creteil.fr

dns_testnames : Conserver les valeurs par défauts ou indiquer ns1.nic.fr (pas la peine d'aller aux Amériques pour cela)

logfile_rotate : Pour faire tourner vos logs et garder un nombre de copies. par défaut 10. attention si votre cache est très utilisé il peut générer un grand volume de logs, pensez donc à réduire ce nombre.

error_directory : Pour avoir les messages d'erreurs en français (indiquer le répertoire où ils se trouvent). Par défaut les messages sont en anglais si vous avez utilisé un rpm. Si vous utilisez un fichier .tar vous devez le compiler avec l'option --enable-err-language=French, afin de les avoir tout de suite en français.

Je pense avoir donné les grandes lignes de la configuration du fichier squid.conf. Voir le fichier squid.conf pour un établissement imaginaire.

9 Accélérer les requêtes

A partir de la version 2, squid dispose d'un mode HTTP-accelerator. Il s'agit en fait d'un cache inversé qui va stocker les données envoyées par l'utilisateur vers le serveur web.

Il faut pour cela mettre dans le fichier squid.conf les lignes suivantes :

```
http_port 8080
httpd_accel_host virtual
httpd_accel_port 80
httpd_accel_with_proxy on
httpd_accel_uses_host_header on
```

Il existe aussi un certain nombre de produits complémentaires qui permettent d'accélérer votre cache. Par exemple en récupérant les pages pendant les heures creuses. Pour plus d'information voir le site de Toulouse.

D'autres paramètres interviennent pour améliorer la rapidité de votre système de cache.

Vous pouvez par exemple installer sur votre machine Squid un cache DNS, augmenter la mémoire sur votre machine.

Enfin il existe un produit BoostWeb (payant) qui compresse les fichiers envoyés aux clients.

10 Contrôler les accès

Pour contrôler tout ce qui passe par votre cache vous devez utiliser les ACL. Elles ont donc deux grandes fonctionnalités : contrôler qui a le droit d'utiliser votre cache et les requêtes que vous avez le droit de faire.

Pour interdire certains sites on peut utiliser les ACL (Access Control List). On peut interdire en fonction du domaine, du protocole, de l'adresse IP, du numéro de port, d'un mot, on peut aussi limiter sur une période.

La syntaxe d'une ACL est la suivante :

acl	aclname	acltype	string[string2]
http_access	allow deny	[!]aclname	
icp_access	allow deny	[!]aclname	

acltype peut prendre comme valeur :

src (pour la source) : indication de l'adresse IP du client sous la forme adresse/masque. On peut aussi donner une plage d'adresse sous la forme adresse_IP_debut-adresse_IP_fin

dst (pour la destination) : idem que pour src, mais on vise l'adresse IP de l'ordinateur cible.

srcdomain : Le domaine du client

dstdomain : Le domaine de destination.

url_regex : Une chaîne contenu dans l'URL (on peut utiliser les jokers).

urllpath_regex : Une chaîne comparée avec le chemin de l'URL (on peut utiliser les jokers).

proto : Pour le protocole.

Exemples :

Interdire l'accès à un domaine : Supposons que nous souhaitons interdire l'accès à un domaine (par exemple le domaine pas_beau.fr). On a donc

```
acl      veuxpas      dstdomain      pas_beau.fr
http_access  deny      veuxpas
http_access  allow      all
```

La dernière ligne ne doit exister qu'une fois dans le fichier squid.conf.

Interdire l'accès aux pages contenant le mot sexe.

```
acl      sexe      url_regex      sexe
http_access  deny      sexe
http_access  allow      all      (Une seule fois à la fin de vos ACL).
```

Attention url_regex est sensible aux majuscules/minuscules. Pour interdire SEXE il faut aussi ajouter SEXE dans votre ACL. Il n'est pas besoin de réécrire toute l'ACL (ce qui serait vite épouvantable) on peut ajouter SEXE derrière sexe en laissant un blanc comme séparation (cela correspondant à l'opérateur logique OU).

On peut placer un nom de fichier à la place d'une série de mots ou d'adresses, pour cela donner le nom de fichier entre guillemets. Chaque ligne de ce fichier doit contenir une entrée.

URL interdites

```
acl url_interdites url_regex "/usr/local/squid/etc/denied_url"
http_access deny url_interdites
```

Des produits associés à Squid permettent un contrôle plus simple (Squid n'a pas vocation à être un moyen de contrôler de nombreux sites). Voir le site de [l'université de Toulouse](http://www.univ-toulouse.fr/~squid/) pour plus d'informations.

SquidGuard permet d'interdire des milliers de sites. La base est entretenue par Toulouse.

Pensez si vous utilisez SquidGuard à configurer la ligne suivante dans le fichier squid.conf :

```
redirect_program /usr/local/squid/bin/SquidGuard
```

Pour contrôler qui a le droit d'utiliser votre cache crée une ACL du type :

```
acl mon_lycee src 192.168.0.0/255.255.0.0
http_access allow localhost
http_access allow mon_lycee
http_access deny all
```

Vous pouvez ajouter autant d'ACL que vous le souhaitez. Par exemple si vous avez plusieurs cartes réseau. 192.168.0.0/255.255.0.0 peut être modifié en fonction de votre plan d'adressage, éventuellement réduit en fonction de votre plan d'adressage (joué sur le masque).

Je ne parle pas ici de toutes les ACL possibles, le principe reste de même que pour les précédentes.

11 Contrôler les accès par authentification

Parmi les demandes qui reviennent le plus souvent, la question de l'utilisation de Squid pour contrôler qui a le droit d'aller sur internet, est l'une des plus fréquente.

On peut imaginer deux solutions :

La première consiste à contrôler les accès par salle et par horaires, en fonction d'un plan d'adressage de

votre établissement. Le travail de l'académie de Grenoble avec [Slis](#) permet de faire cela. On l'administre avec une interface Web. Ce n'est alors pas Squid qui est utilisé pour cela mais le routage.

La deuxième solution est de contrôler en fonction des individus. Squid permet de faire cela, mais la gestion (en l'absence d'une interface) n'en est pas simple. Vous devez utiliser des outils externes pour cela. Il existe aussi des outils Squid LDAP, mais faute d'avoir essayé je ne peux vous en dire plus. Vous pouvez essayer [WPM](#) un produit non complet mais qui vous permet déjà d'interdire certaines choses.

12 Interface web de Squid

Squid dispose en standard d'un script cgi qui donne des informations sur l'utilisation du cache. Pour cela il faut avoir Apache et placer `cachmgr.cgi` dans `cgi-bin`.

Pour configurer complètement `cachmgr.cgi` il faut placer dans le fichier `Sqid.conf` une ligne commençant par `cachmgr_passwd`. Si cette ligne n'existe pas dans le fichier de configuration de Squid alors toutes les fonctions sont permises sauf la modification du fichier `Squid.conf` ainsi que le Shutdown du proxy.

La ligne doit ressembler à :

```
cachmgr_passwd      mot_de_passe      shutdown
```

Si vous souhaitez tout autoriser indiquer `all` à la place de `shutdown`, comme ceci.

```
cachmgr_passwd      mot_de_passe      all
```

Je ne trouve pas personnellement que cette interface apporte grand chose, mais bon. Je vous conseille d'en protéger l'accès (placer un mot de passe au niveau du répertoire en utilisant le serveur Apache).

Il existe des produits que l'on trouve librement sur internet capable de vous donner des statistiques plus claires que `cachmgr`. Je vous propose `prostat` qui est celui que j'utilise. Il est en français et est très clair.

13 Interface web pour configurer Squid

On me pose souvent cette question, je ne sais pas vraiment, essayez Webmin, une interface HTML pour administrer Linux et quelques grands classiques, mais bon ...!

14 Comprendre les logs :

On peut surveiller les logs à l'aide de la commande `tail -f access.log` (les logs de Squid se trouvant dans le répertoire `/var/log/squid` ou `/usr/local/squid/log`).

`access.log` donne les informations sur les requêtes qui ont transité par Squid.

`cache.log` informe sur l'état du serveur lors de son démarrage.

`store.log` informe sur les objets stockés dans le cache.

Voici les valeurs que vous pouvez voir dans le fichier `access.log`. Les codes commençant par `TCP_` sont les requêtes sur le port 8080:

TCP_HIT	Une copie valide se trouve dans le cache
TCP_MISS	Pas dans le cache
TCP_REFRESH_HIT	Objet dans le cache, mais périmé. Squid demande au serveur d'origine si une nouvelle version est disponible, la réponse étant pas de nouvelle version
TCP_REF_FAIL_HIT	Objet dans le cache, mais périmé. Squid demande une mise à jour, mais n'obtient pas de réponse du serveur. Il renvoie alors l'ancienne version.
TCP_REFRESH_MISS	Objet dans le cache, mais périmé. Squid demande une mise à jour qu'il reçoit
TCP_CLIENT_REFRESH	Le client envoie une requête avec une demande de ne pas utiliser le cache. Squid forward la requête.
TCP_IMS_HIT	Le client demande une mise à jour, et l'objet est dans le cache et est récent. Squid ne forward pas la requête.
TCP_IMS_MISS	Le client demande une mise à jour. Squid forward la requête.
TCP_SWAPFAIL	Problème de swap. L'objet semble être dans le cache mais n'est pas accessible. La requête est forwardée.

TCP_DENIED	Accès est dénié.
------------	------------------

UDP_ sont des codes sur le port ICP:

UDP_HIT	Une copie récente de la copie est dans le cache
UDP_HIT_OBJET	Idem que UDP_HIT, mais l'objet est envoyé dans un paquet UDP
UDP_MISS	Objet pas dans le cache ou périmé
UDP_DENIED	Accès interdit pour cette requête
UDP_INVALID	Requête invalide
UDP_MISS_NOFETCH	La requête n'a pas été faite à temps (arrêt ou démarrage du serveur)

ERR_ sont des codes d'erreurs. Ils sont trop nombreux pour être traités ici.

Codes Hiérarchiques

DIRECT	Squid forward directement la requête au serveur d'origine
FIREWALL_IP_DIRECT	Squid forward la requête directement au serveur d'origine, parce que le serveur d'origine est derrière un Firewall
FIRST_UP_PARENT	Squid forward la requête au premier parent disponible de la liste
LOCAL_IP_DIRECT	Squid forward directement la requête au serveur d'origine car l'adresse correspond à une adresse locale
SIBLING_HIT	Squid forward la requête à un cache sibling qui a envoyé un UDP_HIT
NO_DIRECT_FAIL	Squid ne peut pas forwarder la requête parce qu'un firewall l'interdit ou il n'y a pas de cache parent.
PARENT_HIT	Squid forward la requête à un cache parent qui a envoyé un UDP_HIT
SINGLE_PARENT	La requête est forwardée à un cache parent approprié pour cette requête. Il faut que le single_parent_bypass soit actif.
SOURCE_FASTEST	Squid forward la requête au serveur d'origine car la requête à ce serveur source_ping arrive plus vite.
PARENT_UDP_HIT_OBJ	Squid reçoit la réponse dans un UDP_HIT_OBJ du cache parent.
SIBLING_UDP_HIT_OBJ	Squid reçoit la réponse dans un UDP_HIT_OBJ du cache sibling.
DEFAULT_PARENT	Squid forward la requête à un cache parent par défaut sans envoyer une requête ICP avant.
ROUNDROBIN_PARENT	Squid forward la requête à un cache parent round-robin, sans envoyer de requête ICP avant
CLOSEST_PARENT_MISS	Squid forward la requête à un cache parent Cela existe que si vous avez déclaré query_icmp dans le fichier de configuration.
NONE	Squid ne forward aucune requête

15 Contrôler le cache :

- 1) Réinitialiser le cache, pour cela il faut relancer Squid avec l'option -z.
- 2) Purger le cache, pour cela il faut mettre dans le fichier squid.conf les ACL suivantes afin de n'autoriser cela que depuis la machine serveur.

```
acl PURGE method purge
```

```
acl localhost src 127.0.0.1
```

```
http_access allow purge localhost
```

```
http_access deny purge
```

Il suffit alors d'utiliser le programme client avec la syntaxe suivante

```
client -m purge http://le_truc_que_je_veux_enlever.fr
```

On doit avoir alors le message HTTP/1.0 200 Ok

Date : ce_jour

Server: Squid/2.2.4

Si cela ne marche pas vous aurez le message 404 Not Found.

3) Purger la totalité du cache

Il est nécessaire de purger régulièrement la totalité de votre cache afin d'enlever certains fichiers qui pour des raisons diverses refusent de se remettre à jour. Mettez en place une crontab pour faire cela.

16 Installer une interface pour traiter les logs

Je vous propose ici Prostat. Ce produit est en français, et de plus il est assez simple à installer.

La version actuelle est la 1.32 que l'on peut télécharger [ici](#).

Décompresser le fichier gunzip prostat-1.32.tar.gz puis lancer tar xf prostat-1.32.tar.

Vous allez avoir un sous répertoire prostat-1.32 avec un certains nombre de fichiers, Lire le fichier LISEZ_MOI (A vous de voir ou vous souhaitez l'installer).

Il faut avoir installé le compilateur C sur votre machine (et les bonnes librairies) pour pouvoir le compiler ainsi que make (par défaut sur la Mandrake cela n'est pas installé).

Se placer dans le sous répertoire gd1.2 puis compiler à l'aide de la commande make, puis se placer dans le répertoire prostat-1.32 et configurer les fichiers analhead.h et prostat.conf.

Une fois compilé, vous pouvez toujours modifier le fichier prostat.conf. Lancer alors la commande make pour compiler prostat.

Lancer prostat à l'aide de la commande indiquée dans LISEZ_MOI et de cron.

crontab -e (vous ouvrez alors une fenêtre vi) puis pour avoir les logs tous les dimanche à 2 heures du matin indiquer 0 2 * * sun /usr/bin/prostat-log (prostat-log étant votre fichier ou se trouve le script qui se trouve dans LISEZ_MOI.. (Voir le résultat sur le site de [Jussieu](#)).

17 Caches hiérarchiques

Squid est capable de "discuter" avec d'autres proxy cache, à l'aide de ICP (Internet Cache Protocole) sur un port particulier (3130 par défaut). On peut ainsi cascader un certain nombre de caches.

Il faut pour cela dans le fichier de configuration de Squid valider la ligne

```
cache_peer      type      http_port      icp_port      options
```

Pour le type on a comme possibilité :

parent : Le cache ainsi défini contacte le cache parent et attend la réponse via ICP, si le cache parent ne possède pas la réponse il se la procure et la fait suivre au cache demandeur (votre cas si vous passez par Oléane ou par le rectorat).

sibling : Le cache contact le cache parent, si celui ci n'a pas la réponse, le cache enfant se procure directement la page. Celle ci n'est pas chargée sur le cache parent.

Pour http_port et icp_port utiliser 8080 et 3130, mais cela n'est pas obligatoire.

Pour les options on dispose des options suivantes :

proxy-only : On indique à l'aide de cette option que les données envoyées par le proxy interrogé ne sont pas dans le cache local.

weight : pour avoir une pondération entre plusieurs caches

no-query : Afin de ne pas envoyer de requêtes ICP au proxy indiqué.

round-robin : Utiliser pour répartir la charge entre plusieurs proxy.

Ainsi pour l'académie de Créteil il faut indiquer :

```
cache_peer      proxy.ac-creteil.fr parent 8080 3130 no-query default
```

```
Indiquer aussi cache_peer_domain proxy.ac-creteil.fr !votre_lycee.ac-creteil.fr
```

Afin de ne pas envoyer les requêtes pour votre établissement au cache parent.

Ce proxy n'est utilisable que si vous êtes un établissement scolaire connecté par le rectorat ou par Oléane.

Il existe une autre option de caches hiérarchiques qui est : cache_digests. Pour pouvoir l'utiliser il faut

avoir compilé squid avec l'option `./configure --enable-cache-digests`. Cette solution est plus intéressante que ICP, car le trafic entre les caches est plus réduit. Elle n'a pas d'intérêt pour un établissement scolaire.

Il faut alors mettre la ligne suivante : `cache_peer mon_premier_cache.fr sibling 8080 3130 no-query proxy-only`

Dans le cas des caches hiérarchiques indiquer plutôt l'adresse IP à la place de `mon_premier_cache.fr`

18 Répartition de charge entre plusieurs caches

Il existe plusieurs solutions pour cela.

- 1) Utiliser la configuration automatique des clients. Voir pour cela [configuration automatique des clients](#)
- 2) Utiliser un script perl à la place d'un fichier .pac capable en fonction des adresses des clients de dispatcher les requêtes.
- 3) Utiliser le DNS (round robin).
- 4) Utiliser un L4 switch capable de faire cela (load balancing).
- 5) Mettre en place une solution à l'aide d'une machine sous Linux capable de dispatcher les demandes (par exemple Linux IPVS).
- 6) Mettre en place un Squid sans cache pour répartir la charge, avec comme cache parent l'option parent.

19 Configurer les clients

Pour configurer les clients, on peut utiliser la configuration manuelle ou la configuration automatique.

Voir [ici pour plus de détails](#)

20 Forcer le passage par SQUID

Il existe plusieurs solutions:

- 1) Configurer votre navigateur avec le bon proxy ou en utilisant le fichier de configuration automatique et le rendre impossible à changer. Mais cela nécessite que vous contrôliez les clients ce qui n'est pas toujours le cas.
- 2) Placer deux cartes réseau dans votre machine avec deux plans d'adressage distinct (vos utilisateurs ne connaissent pas l'adresse de votre routeur et de la deuxième carte réseau), seul alors Squid est capable d'aller sur internet. Cette solution me semble la plus efficace.
- 3) Compléter Squid avec l'option `--enable-ipf-transparent` (Voir la doc de Squid).
- 4) Intercepter les requêtes sur le port 80 pour les rediriger sur Squid. Utiliser pour cela [TransProxy](#)
- 5) Il existe maintenant des Switchs capables de faire cela mais le prix reste élevé.

21 Dans votre établissement :

Dans l'établissement "qui est le votre" la configuration de Squid peut être la suivante.

Une machine avec un disque assez grand (mais la taille des disques actuels et le prix ne posent plus de problèmes, un SCSI plutôt d'un IDE). Prévoir assez de mémoire sur la machine.

Je propose une RedHat ou Mandrake qui sont des distributions assez simples à mettre en oeuvre. De plus on trouve dans beaucoup de revue la Mandrake. Utiliser une version récente de Squid (version 2 stable 4 que l'on peut trouver sur tous les CD, mais aussi sur le [site de Squid](#)).

Vous trouverez ici un [fichier Squid.conf](#) pour un établissement de l'académie, et [ici](#) un fichier pour la configuration automatique de vos clients.

Le paramétrage des fichiers tient compte de la présence ou non d'un serveur Web local (dans l'établissement) et de l'utilisation du proxy cache du rectorat comme cache parent.

22 Sources d'information sur Squid :

Le site de Squid : <http://squid.nlanr.net/>

Université de Toulouse : <http://cache.univ-tlse1.fr/documentations/cache/index.html>

Le cru : <http://www.cru.fr/renater-cache/er-cache/>



Samba : compléments

Source de documentation

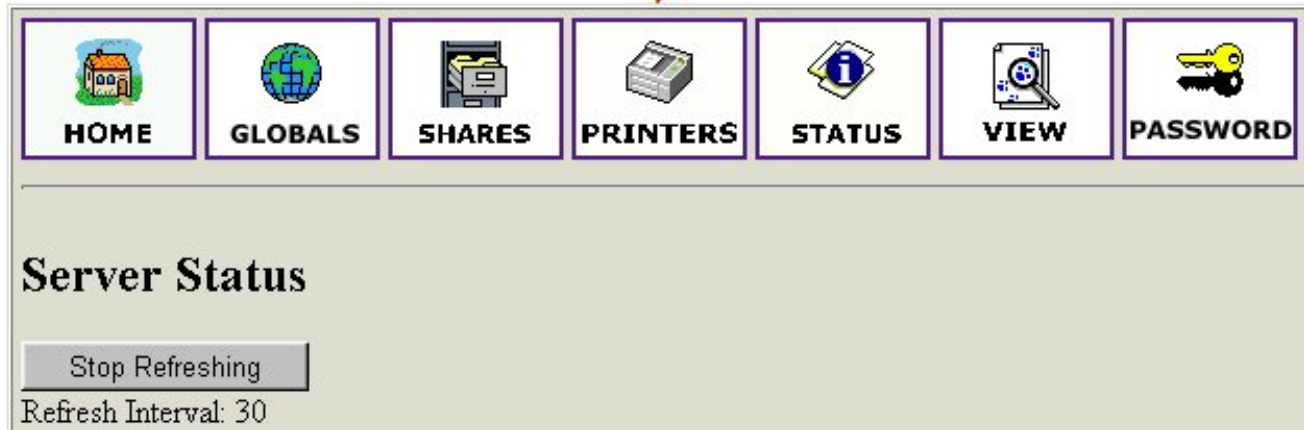
- Un livre : *Samba, installation et mise en oeuvre*, édition O'Reilly (août 2000)
- Référence : les documents suivants fournis avec Samba et présents dans le répertoire `/usr/doc/samba-1.9.18p10/docs` :
DIAGNOSIS.txt, DOMAIN.txt, DOMAIN_CONTROL.txt, ENCRYPTION.txt, Win95.txt, Win95_PlainPassword.reg
- site [consacré à samba](#)

Diagnostic

- Rappel : avant toute manipulation hasardeuse sur les fichiers de configuration, en effectuer une copie de sauvegarde.
- En cas de problème non résolu, pour cerner la difficulté, il est conseillé d'effectuer les 10 tests systématiques de `DIAGNOSIS.txt`

SWAT : interface d'administration

- **swat** (= Samba Web Adminstrating Tool) est un utilitaire permettant l'administration de Samba, via une interface Web sur un poste client. Il est normalement installé en même temps que `samba`.
- Pour se connecter à `swat`, on passe une requête au serveur au port 901 du genre :
http://p00:901, si p00 est le nom du serveur (à défaut, utiliser son adresse ip).
En cas d'erreur, *serveur absent sur le port 901*, il faut effectuer :
 1. Ouvrir le fichier de configuration de `tcp/ip`, `/etc/inetd.conf` et décommenter la (dernière) ligne : `swat stream tcp nowait.400 root /usr/sbin/swat swat`
 2. Vérifier dans le fichier **`/etc/services`** la présence de la ligne **`swat 901/tcp`**
 3. Il faut redémarrer le démon **`inetd`** par `/etc/rc.d/init.d/inet restart`
 4. Vérifier que le port 901 est alors bien accessible.
- Expérimenter les différents paramétrages qu'on peut effectuer comme `root` ou simple user.
Mais ATTENTION ! toute validation de modification de `smb.conf` effectuée sous l'interface SWAT provoque la réécriture complète du fichier en l'épurant de toutes lignes superflues, y compris les COMMENTAIRES (qui ne sont pas superflus ...). Par conséquent, il faut faire une copie du `smb.conf` AVANT toute intervention comme `root`. On pourra ensuite la restaurer.



WEBMIN

- C'est un utilitaire complet de gestion de Linux et de ses services, écrit en Perl. L'administrateur peut agir par l'intermédiaire d'une interface WEB. Le module Samba permet de gérer les partages et les utilisateurs Samba, donc possède des fonctionnalités bien plus étendues que SWAT.
Vivement une version en français et une aide !
- Pour l'installation et la configuration, [voir ce document](#)

[Index de Webmin](#)
[Configuration du](#)
[Module](#)

Samba Share Manager

Share Name	Path	Security
homes	<i>All Home Directories</i>	Read/write to all known users
netlogon	/home/netlogon	Read/write to root,smbadmin Read only to all other known users
printers	<i>All Printers</i>	Printable to all known users
public	/home/tmp	Read/write to everyone
stagiaire	/home/rep-stagiaire	Read/write to all known users
cdrom	/mnt/cdrom	Read only to everyone
logiciel	/appli	Read only to everyone
web	/home/httpd/	Read/write to all known users

[Create a new file share](#)
 [Create a new printer share](#)
 [Create a new copy](#)
 [View All Connections](#)

Le problème des mots de passe Samba

I. Mise en oeuvre pour Windows98 (sans explication, si on est pressé ...)

1. Activer le cryptage dans `smb.conf` : décommenter ces 2 lignes, puis redémarrer Samba

```
encrypt passwords = yes
smb passwd file = /etc/smbpasswd
```

2. Le fichier `/etc/smbpasswd` n'est pas créé lors de l'installation.
Il le sera lors de l'ajout du premier compte Samba :

```
smbpasswd -a jean
---> le fichier est créé avec le mot de passe crypté
```

3. L'utilisateur `jean` peut maintenant se connecter à son répertoire personnel.

II. Connexion sous les "anciens" Windows95

Que l'on active ou non le service de cryptage dans `smb.conf`, et quel que soit le positionnement de `EnablePlainTextPassword` dans la base de registre des clients 95, apparemment cela n'a pas d'incidence sur les connexions de ces stations Windows 95 (Reste à comprendre pourquoi).

III. Connexion sous Windows98

Une station 98 correctement configurée (domaine MS, NetBios activé) voit le serveur Samba, mais le processus de connexion n'aboutit pas ...

MS a modifié l'authentification des mots de passe. Par défaut, Windows98 les envoie maintenant cryptés par défaut et le serveur Samba les attend en clair ... Il faut donc intervenir pour rétablir une cohérence.

Il y a 2 solutions :

IV. On ne veut pas crypter les mots de passe

S'assurer d'abord sur le serveur que Samba ne crypte pas

```
encrypt passwords = no
```

Puis intervention dans la base de registre pour que Windows arrête de crypter

1. Lancer `c:\windows\regedit.exe`
2. Dans la base de registre, se placer dans le rép.
`HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\VxD\VNETSUP`
3. Menu Edition / Nouveau / valeur DWORD
4. Une nouvelle entrée est créée, lui donner le nom `EnablePlainTextPassword`.
5. Double-cliquer pour l'éditer, et donner la valeur **hexadécimale 1**.
6. Fermer et redémarrez la machine .. et ça fonctionne !

Test sur station 98

Avec `EnablePlainTextPassword="dword:00000001`, le service de cryptage n'est pas utilisé et il y a connexion possible pour les comptes existants sur le serveur Linux

V. On paramètre Samba pour le cryptage des mots de passe

Dans la section `[global]` du fichier `/etc/smb.conf`, on doit trouver les lignes suivantes :

```
# You may wish to use password encryption. Please read
# ENCRYPTION.txt, Win95.txt and WinNT.txt in the Samba documentation.
# Do not enable this option unless you have read those documents (!)
encrypt passwords = yes
smb passwd file = /etc/smbpasswd
```

Les mots de passe cryptés utilisés par Samba seront donc placés et lus dans le fichier `/etc/smbpasswd` qui appartient à `root`.

Le rôle et la gestion de `/etc/smbpasswd` sont expliqués dans le fichier `ENCRYPTION.txt`

La création d'un compte utilisateur s'accompagne de l'ajout de la ligne correspondante dans `passwd` et dans `shadow`, mais n'opère pas de modification dans `smbpasswd`, si on a créé le compte "à la main" avec la commande `useradd`.

Pour crypter les comptes samba qui ne l'auraient pas été, il faut effectuer les opérations suivantes :

1. activer le cryptage dans `/etc/smb.conf`
2. créer le fichier `smbpasswd` en récupérant les comptes Linux
3. activer les mots de passe Samba
4. paramétrer les clients 95/98 pour activer le cryptage des mots de passe

VI. Cryptage par smbpasswd et par linuxconf

- Lorsque les comptes ont été juste créés par la commande `useradd`, aucun mot de passe n'a encore été défini et chiffré. Pour s'en convaincre, examiner `/etc/shadow` : une entrée a juste été créée au nom du compte.
- On définit le mot de passe comme d'habitude, avec l'utilitaire `passwd`. Alors, `root` peut constater qu'un enregistrement contenant le mot de passe crypté a été ajouté au fichier `/etc/shadow`; par contre, ni cryptage, ni ajout d'une entrée n'ont été effectués dans `/etc/smbpasswd`
- La commande **`smbpasswd nom`** demande 2 fois le mot de passe samba du compte dont on fournit le nom.
Si
- `smbpasswd` crypte dans `/etc/smbpasswd`, avec un algorithme, bien sûr identique à celui que les stations Windows utilise. Lors du premier accès à un partage situé sur le serveur Samba, le mot de passe envoyé chiffré par la station est comparé à celui résident dans `/etc/smbpasswd`. Cette comparaison détermine la permission ou non d'y accéder, avec les droits attachés au partage.
- L'utilitaire `linuxconf` enchaîne et synchronise les 2 mots de passe, apparemment sans qu'on le lui demande !

VII. Cryptage par Samba, exemple de procédure

1. Le cas échéant, activer le cryptage dans le fichier `smb.conf`
2. Sauvegarde du fichier `smbpasswd` existant : `cp smbpasswd smbpasswd.old`
3. Création d'un nouveau fichier `smbpasswd`, à partir du fichier `passwd`, de façon à récupérer tous les comptes déjà créés, avec l'utilitaire `/usr/bin/mksmbpasswd.sh`
`cat /etc/passwd | mksmbpasswd.sh > /etc/smbpasswd`
4. Examiner le résultat avec `less /etc/smbpasswd` : les mots de passe n'existent pas encore, seules les lignes correspondant aux comptes linux sont créées
Vérifier la protection 644 de `smbpasswd`
5. Créer un mot de passe samba pour `stagex`, avec la commande **`smbpasswd -a stagex`**
6. Aussitôt et sans nécessité à se reconnecter, `stagex` peut accéder aux ressources partagées. Vérifier la présence du mot de passe dans `/etc/smbpasswd`
7. Pour déprotéger un compte Samba, éditer `/etc/smbpasswd` et remplacer les 11 premiers 'X' du mot de passe par la chaîne "NO PASSWORD".
8. Sur les stations Windows
Sur les stations Windows 98, le cryptage des mots de passe est activé par défaut, il n'y a donc pas à intervenir.
En cas de problème, notamment sur des Windows 95 "anciens", ouvrir la base de registre et changer la clé suivante :
`HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\VxD\VNETSUP`
`"EnablePlainTextPassword"=dword:00000000`

VIII. La synchronisation des mots de passe

Il s'agit d'une procédure permettant de garder une cohérence entre les mots de passe des comptes Linux et Samba lorsque l'un d'entre eux est changé depuis une station.

pour demander la synchronisation

`unix password sync = Yes`

pour indiquer la commande linux pour modifier le mot de passe

`passwd program = /usr/bin/passwd %u`

indique que Samba doit renvoyer 2 fois %n (nouveau mot de passe) au

programme de traitement des mots de passe (indiqué par passwd program)

en réponse à la demande "new password"

`passwd chat = *new*password* %n\n *new*password* %n\n *succes*`

Mini FAQ Samba

Quelques erreurs

- Le serveur Samba n'est pas visible d'une station
 - Les machines sont-elles dans le même plan d'adresse IP ?
 - Les noms de domaine WorkGroup sont-ils synchronisés ?
 - A t-on bien redémarré les tâches `smdbd` et `nmbd` après une modification dans `smb.conf` ?

- Le serveur est "vu" dans le voisinage réseau mais une tentative d'accès se solde par le message :
"l'ordinateur ... n'est pas accessible, l'ordinateur spécifié n'a pas reçu votre requête ..." ou *"\p00 n'est pas accessible, le réseau est occupé"* (!)
 - Vérifier dans `smb.conf` que la ligne **hosts allow** est correcte, si elle est présente, par exemple : `hosts allow = 10.194.2. 127.` pour un sous-réseau d'adresse 10.194.2.0
 - On a constaté la reprise de la connexion, en commentant cette ligne, et en décommentant la commande `interfaces 10.194.2.100/24` (en supposant que l'adresse IP du serveur est 10.194.2.100)
- On reçoit des messages de log sur la console du serveur (quand on "restart" le démon Smb)
 Vérifier que la taille maximum du fichier de log `/var/log/samba/log.nmb` n'est pas atteinte. Cette taille est fixée par défaut à 50 Ko dans `smb.conf`, section `[global]`

Station Linux, cliente d'un serveur Windows9x

Un client SMB pour un hôte Linux est inclus dans Samba.

La connexion étant établie, on communique avec le serveur Windows à travers une interface semblable au ftp, ce qui permet de faire du transfert de fichiers.

Connexion au serveur Windows

La connexion est établie par la commande `/usr/sbin/smbclient -L nom` où on utilise le nom NetBIOS de la machine Windows.
 Un mot de passe est requis s'il y a une protection, sinon il suffit de valider.

Exemple :

```
$ smbclient -L pc1
```

```
Password:
```

Sharename	Type	Comment
-----	----	-----
CD_PC1	Disk	
C_PC1	Disk	
D_PC1	Disk	
I_JP150	Printer	
PRINTER\$	Disk	
IPC\$	IPC	Communication entre processus distants

Server	Comment
-----	-----
Workgroup	Master
-----	-----

Transfert de fichiers

La commande `smbclient \\\\pc1\\C_PC1` permet d'accéder sur PC1 à la ressource partagée, nommée ici C_PC1 (ie le disque C:). Il faut bien entendu respecter les noms de partage attribués sur le serveur WorkGroup.

On obtient un prompt semblable à ftp. Pour connaître les commandes :

```
smb: \> h
ls          dir          lcd          cd           pwd
get         mget         put          mput        rename
more        mask         del          rm           mkdir
md          rmdir        rd           prompt      recurse
translate   lowercase   print        printmode   queue
cancel      stat         quit         q           exit
newer       archive     tar          blocksize   tarmode
setmode     help        ?            !
smb: \> cd excel
smb: \excel\>
```

Montage de répertoire Windows

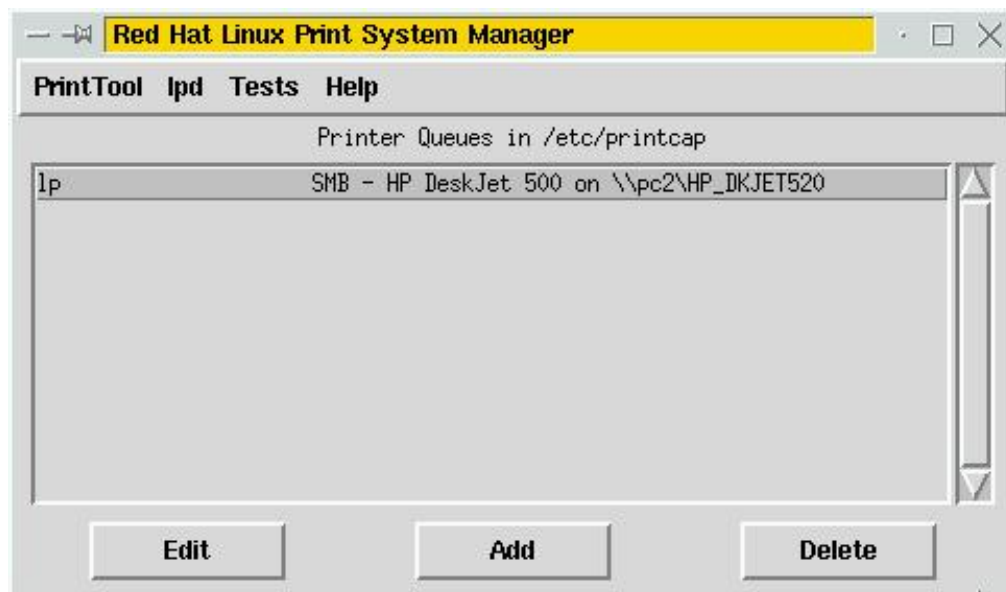
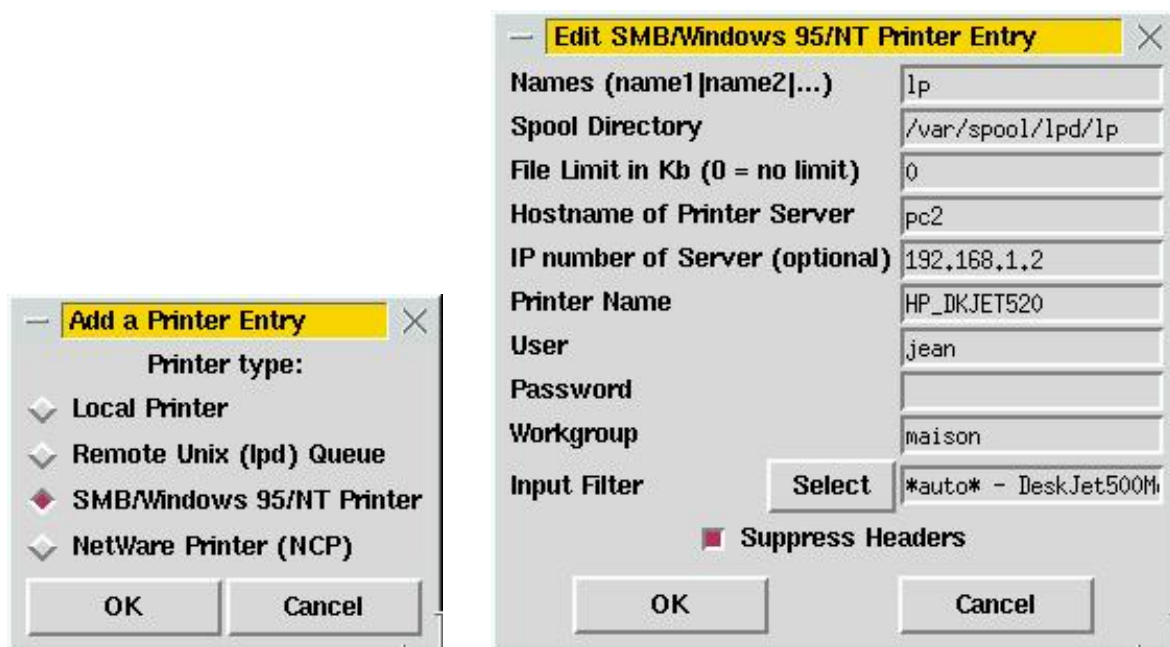
- La commande **smbmount** permet de monter (comme par NFS) des répertoires distants d'une machine Windows9x, sur l'arborescence Linux, et ainsi de disposer des fichiers.
 - Pour monter la ressource `//PC1/C_PC1` sur le répertoire `/mnt/diskc_pc1` du système de fichiers :
smbmount //PC1/C_PC1 /mnt/diskc_pc1
 Password : valider.
 On parcourt ensuite le répertoire `/mnt/diskc_pc1` comme un dossier local.
 - S'il y a blocage (faire ctrl-C) et se demander :
 Le client Samba connaît-il la machine nommée PC1 ?
 --> il est nécessaire que le nom de l'hôte (ici PC1) ait été déclaré dans `/etc/hosts`
 La ressource C_PC1 est-elle déclarée partagée sur le serveur Windows9x ?
 --> clic-droit dans l'explorateur sur le disque concerné/partager ../ etc..
 - Le client Samba respecte le type de partage Windows en lecture seule ou lecture/écriture.
 Supposons que C:\temp soit un sous-répertoire du disque C: du serveur WorkGroup PC1
C: est partagé en lecture seule sous le nom de partage C_PC1
 >C:\temp est partagé en lecture/écriture sous le nom de partage C_temp_PC1
Monter les ressources :
`smbmount //PC1/C_PC1 /mnt/diskc_pc1`
`smbmount //PC1/C_temp_PC1 /mnt/temp_pc1`
Tester :
 Transférer des fichiers dans C:\temp
 Par exemple, copier `/etc/fstab` dans `/mnt/diskc_pc1` et dans `/mnt/temp_pc1`.
 - Pour partager les fichiers montés en lecture-écriture pour tous, on ajoute le paramètre **-f 777**
 - Pour démonter la ressource Windows, utiliser `subumount point-montage :`
sbumount /mnt/diskc_pc1
-

Utiliser une imprimante partagée par Windows

Soit une imprimante lointaine pour la station Linux, installée sur un serveur Windows9x.

Voyons comment cette ressource peut être utilisée par le client Linux.

- Soit une imprimante sur l'exemple : nommée HP_DKJET520, gérée par le serveur d'impression sur la station Windows pc2, bien sûr déclarée partagée sous ce nom pour le réseau Microsoft
- Sur la station cliente Linux, lancer l'utilitaire **X printtool**.
 Ajouter une imprimante / sélectionner *SMB/Windows 95/NT Printer*
 Paramétrer
 - nom par défaut : lp
 - Hostname : pc2 et numéro IP
 - Printer Name : HP_DKJET520
 - Workgroup : nom donné dans `/etc/smb.conf`
 - Input Filter/Select : choisir le modèle d'imprimante (ici HP DeskJet 500) et effectuer les réglages habituels
- Effectuer un test (menu Tests)
- Utilisation sous Linux, par exemple :
\$ lp /etc/smb.conf





TP permissions d'accès aux fichiers



Ces exercices doivent aider à maîtriser le système de droits de Linux, finalement assez simple.

Il est recommandé de les chercher d'abord "sur papier" puis de tester pour vérifier.

Exercice 1

1. Quels sont les droits sur les répertoires personnels (par exemple `stages`) ?
2. Un utilisateur différent `stagey` peut-il y pénétrer ou seulement lister ses fichiers ? et `totox`, le pourrait-il s'il faisait partie du groupe de `stages` ?
3. Quelles commandes devrait écrire `stages` pour accorder le droit de visite de son rép. perso seulement à `totox` ?

Exercice 2

1. Comparer les permissions de `/etc/passwd` et `/etc/shadow`.
Pourquoi a-t-on nommé ainsi ce dernier fichier ? `stages` peut-il le lire ? et voir sa présence ?
L'examiner pour deviner son rôle.
2. Par précaution, en faire une copie sous le nom `shadow.bak` dans `/home/temp` ! vérifier les droits de `/home/temp/shadow.bak`
3. Pensez-vous tout de même pouvoir supprimer le fichier précédent ? Concluez !
4. `root` fait maintenant une copie de `shadow` chez vous, dans `/home/stages`, sous le nom `shadow.bak` et vous accorde la propriété de la copie.
 - a) Comment fait-il ?
 - b) `stages` vérifie le résultat
5. Vous éditez ce fichier avec Midnight Commander, vous le modifiez, par exemple en supprimant des lignes, et vous faites une mise à jour.
Ecrivez le mode opératoire.
La mise à jour sera-t-elle réalisée ? pourquoi ?
6. Pensez-vous que `stages` puisse supprimer ce fichier ?
Essayez et expliquez !

Exercice 3

1. En tant que **stagex**, pouvez vous créer le rép. temporaire `/home/temp` ? essayez ! pourquoi ?
2. Effectuez cette création comme **root** (pensez à la commande `su`).
3. Accorder les permissions maximales sur `/home/temp`; vérifiez.
4. **totox**, toujours lui, tout content d'avoir enfin un droit d'écriture, dans `/home/temp` essaie de copier les 2 fichiers système `/etc/hosts` et `/etc/passwd` dans `/home/temp` ? y parviendra t-il ? pourquoi ? que donne `[totox@p0x ] ll /home/temp` ?
5. **totox**, essaie maintenant de supprimer ces 2 fichiers de `/etc`. Réussit-il ?
6. Effrayé à l'idée de se faire pincer par le (ou la) redoutable **root**, **totox** veut masquer sa faute tout en faisant punir **stagex** à sa place ! Pour cela, il veut que **stagex** devienne propriétaire du fichier copié `passwd`. Comment s'y prend t-il ? Réussit-il ? Et vous comment auriez vous fait ?

Exercice 4

Il s'agit de créer un rép. partagé par tous les membres **stagex** du groupe **stagiaire**. Normalement, ce groupe a déjà été créé et rempli de comptes **stagex**.

1. Créez dans **/home** un répertoire appelé **rep-stagiaire**. Rappelez pourquoi cette tâche relève des prérogatives de **root**
2. Faites-le appartenir au groupe **stagiaire**
3. Modifier les permissions sur le rép, pour que tous les membres du groupe **stagiaire** puissent y écrire et s'y déplacer.
4. En tant que **stagex**, vous créez un fichier, par exemple un petit fichier texte (à l'aide de `vi` ou d'un éditeur graphique comme `kedit`) et vous le déposez dans `/home/rep-stagiaire`. Si vous êtes paresseux, vous y faites une copie d'un fichier qcq, par exemple `/etc/hosts`, mais en attribuant des droits 660

```
[stagex@p0x etc] cp hosts /home/rep-stagiaire
[stagex@p0x etc] chmod 660 hosts
```
5. Vérifier le bon accès en lecture seulement pour les membres du groupe
Ainsi **totox** qui a fini par être exclu du groupe **stagiaire** (surtout après l'exercice 3) ne doit pas pouvoir le lire. A vérifier.
6. Votre collègue (ou votre double !) le perfide **stagey** (`y#x`), tente de supprimer ce fichier ou de le renommer
Y parvient-il ? Essayez !
Pourtant, vérifiez que ce fichier appartient au groupe **stagex**
N'est-ce pas inquiétant ? Expliquez comment cela est possible.
7. Demandez à **root** de positionner le "sticky bit" sur le répertoire partagé.
Vérifiez bien que le problème est réglé et protège le propriétaire des tentatives de suppression ou de changement de nom de ses fichiers.

Prolongement : vérifier que cette protection s'applique aussi à distance sous Samba

Pour voir une [proposition de corrigés](#)



TP permissions d'accès aux fichiers corrigés proposés

Exo 1

1. `drwx----- stagex stagex /home/stagex`
 2. même si `totox` fait partie du groupe `stagex`, le répertoire n'accorde pas de permissions **x** de parcours, ni même **r** de lecture.
 3. Pour que `totox` puisse accéder en lecture au rép. perso. de
 - . ajouter `totox` dans le groupe `stagex`, avec `linuxconf`
Attention ! cela peut être imprudent : en effet, par la suite `totox` pourra user et abuser des droits accordés
 - b. `chmod g+r /home/stagex`
-

Exo 2

1. Il n'est pas caché, `stagex` peut le voir avec `ll /etc/sha*`
Mais il n'est pas lisible : permission non accordée pour `less /etc/shadow` !
2. `root` effectue la copie : `cp /etc/shadow /home/temp/shadow.bak`
3. avec une permission `r--` sur ce fichier, `root` peut quand même le supprimer !
4. `root` effectue la copie : `cp /etc/shadow /home/stagex/shadow.bak`
et accorde la propriété : `chown stage1. /home/stagex/shadow`
5. `stagex` passe les commandes `cd`, puis `ll` et observe :
`-r----- stagex stagex shadow.bak`
6. `stagex` peut lire le fichier par `less shadow.bak` et l'édite avec `mc/F4`, supprime une ligne, et veut sauvegarder par `F2` --> refus et invite à le renommer.
C'est normal, le droit **w** n'est pas positionné !
7. `stagex` veut le supprimer `rm shadow.bak`
Il reçoit l'avertissement :
rm: détruire le fichier protégé en écriture shadow.bak et répond `y`, et ... c'est fait.
En effet `stagex` possède la permission **w** sur son rép perso `/home/stagex`.

Remarque :

Si `stagex` se bloque ce droit, ce ne serait plus possible !

```
cd /home
```

```
chmod 500 stagex
```

```
cd
```

```
rm shadow.bak
```

rm: détruire le fichier protégé en écriture 'shadow.bak'

```
--> y
```

```
rm: Ne peut délier 'shadow.bak' : Permission non accordée
```

Exo 3

1. le rép /home n'accorde un droit d'écriture qu'à root
 2. [stagex@p0x] su
[root@p0x] mkdir /home/temp
 3. chmod 777 /home/temp
 4. cd /home/temp
[totox@p0x temp] cp /etc/hosts . *pour déplacer dans le rep. courant*
 5. Fort heureusement totox n'a pas de droit d'écriture w sur /etc, il ne peut donc rien y supprimer !
 6. chown stagex /home/temp/*
opération non permise !
Pris de panique totox n'a plus pensé qu'il a le droit de supprimer cette copie qui lui appartient ...
-

Exo 4

On suppose la situation initiale suivante :

1. root a créé le rep rep-stagiaire
mkdir /home/rep-stagiaire
2. Il a attribué la propriété collective de ce rép. au groupe stagiaire,
chgrp stagiaire /home/rep-stagiaire
3. avec un accès complet, et rien pour les autres utilisateurs.
chmod 770 /home/rep-stagiaire

Vérification :

```
ll /home
drwxr-xr-x root      root      httpd/
drwx----- stagex   stagex    stagex/
drwx----- stagey   stagey    stagey/
drwxrwx--- root     stagiaire rep-stagiaire/
```

4. stagex, très rétro, crée ce texte directement saisi à la console avec cat
cat, par défaut admet le clavier comme canal d'entrée

```
cd /home/rep-stagiaire
cat > doc-stgx.txt
Voici un petit texte sans prétention,
mais auquel je tiens beaucoup.
Mes collègues peuvent le lire, y répondre
mais je leur demande de ne pas le modifier
```

ni l'effacer. Merci.

signé : stagex

Ctrl-D

Par précaution stagex, pas naïf, en fait une copie privée, sans droit de groupe :

```
cp doc-stgx.txt doc-stgx.sauve
```

```
chmod 600 doc-stgx.sauve
```

```
ll doc-stgx.sauve
```

```
-rx-----
```

5. stagey lit le texte par exemple avec `less /home/rep-stagiaire/doc-stgx.txt`
On vérifie que totox ne fait plus partie du groupe stagiaire et qu'il ne peut pas lire.

```
groups totox
```

```
totox : totox
```

```
[totox@p0x home] less rep-stagiaire/doc-stgx.txt
```

Permission non accordée

6. stagey supprime sans problème le document de stagex

```
[stagey@p0x rep-stagiaire] rm doc-stgx.txt
```

```
rm: détruire le fichier protégé en écriture 'doc-stgx.txt' ?
```

y (répond t-il sournoisement)

```
[stagey@p0x rep-stagiaire] ll (pour vérifier, le traître !)
```

7. Pour n'autoriser les suppressions qu'effectuées par le propriétaire (ou de root), root passe la commande

```
chmod +t /home/rep-stagiaire
```

```
(ou chmod 1770 /home/rep-stagiaire)
```

```
ll /home
```

```
drwxrwx--T root stagiaire rep-stagiaire/
```

(rappel : **T** au lieu de **t**, car pas de droit **x** pour la catégorie *other*)

Ces 2 tentatives échoueront :

```
[stagey@p0x rep-stagiaire] rm doc-stgx.txt
```

```
[stagey@p0x rep-stagiaire] mv doc-stgx.txt doc-stgy.txt
```



TP SAMBA

TP1 mise en oeuvre de la station cliente

1. Préalable

- Pour ce TP, on dispose dans la salle de formation du serveur Samba **p00** de nom Netbios "*Serveur Samba*" et de numéro IP **10.177.200.100**
- Les comptes utilisateurs (*stagex/stgx*) sont supposés créés et cryptés sous Linux et sous Samba.
- Le fichier **smb.conf** d'origine su serveur Samba a été très peu modifié, dans sa section globale :

```
[global]
  workgroup = fctice77
  encrypt passwords = yes
  smb passwd file = /etc/smbpasswd
```

2. Configuration de la station

Reconfigurer complètement les propriétés réseau de la station Windows 98 pour permettre la connexion au serveur Samba.

- Enlever les protocoles ici superflus (*netbeui, ipx/spx*), ajouter *tcp/ip* et *netbios* s'ils sont absents.
- Ajouter, s'il est absent, le client pour réseau Microsoft et enlever éventuellement les autres services actifs (comme *client pour réseau Netware, service NDS*)
- activer *Partage des fichiers*
- Paramétrage réseau
 - identification : **PCx**
 - groupe de travail : **fctice77**, il doit être le même que celui indiqué dans **smb.conf** du serveur **p00**
- Contrôle d'accès au niveau ressources
- Paramétrage protocole TCP/IP
 - adresse IP **10.177.200.(100+x)** où x est le numéro de la station.
 - Netbios activé
 - Eventuellement : *Passerelle vers Internet* et *DNS*
 - Et, inévitablement .. relancer Windows

3. Tests

- Se connecter sous l'identité *stagex/stgx*
- Les 2 machines **p00** et **pcx** doivent apparaitre dans le groupe de travail **fctice77** du voisinage réseau.
- Un double clic sur **p00** doit faire apparaitre le partage nommé **stagex**, qui n'est autre que le répertoire personnel de l'actuel utilisateur situé à **/home/stagex** sur le serveur.
- Une éventuelle icone d'imprimante apparait également, si une imprimante a été installée localement sur

le serveur Linux

- A l'aide de l'Explorateur, copier quelques fichiers dans le répertoire personnel. Tester les droits d'accès en lecture et en écriture. Un autre utilisateur **stagey** peut-il en supprimer ? et même y accéder ?
- Créer un lecteur réseau **P** : dans le poste de travail pour accéder directement au rép. personnel (le chemin doit être indiqué suivant la syntaxe Windows : `\\p00\homes`)

4. Impression avec Samba

- Installer sur chaque station l'imprimante **lp** supposée déjà installée sur le serveur **p00** au fichier de périphérique `/dev/lp0`
- Suivi des jobs d'impression
Simultanément, sur un terminal du serveur, la commande **lpq** renseigne sur ce job d'impression.
Si le serveur est sous X, lancer l'utilitaire **KLpq** d'information et de gestion de la file d'impression par la commande :
K / Utilitaires / File d'impression
- Exemple
[jean@p00 /] lpq
lp is ready and printing

Rank	Owner	Job	Files	Total Size
active	jean	13	/var/spool/samba/pc2.a01799	113879 bytes

TP2 mise en oeuvre du serveur

Chaque groupe doit disposer au moins de 2 machines.

- un serveur Linux opérationnel **p0x** sur lequel il s'agira de configurer l'accès au service Samba, donc le fichier `/etc/smb.conf`
- la station Windows9x **pcy** précédemment configurée
- Celles-ci seront déclarées dans un (sous-)groupe de travail **factice77x**
 - Modifier `/etc/smb.conf`, comme cela a été expliqué au TP1 (1)
Redémarrer le service Samba : `/etc/rc.d/init.d/smb stop | start`
 - Définir un nouvel utilisateur `stagex/stgx`
 - `useradd stagex`
 - `passwd stagex` (mot de passe = `stgx`)
 - **`smbpasswd -a stagex`**, pour crypter le mot de passe Samba dans `/etc/smbpasswd`
 - Tester complètement sous le compte `stagex/stgx`, comme précédemment (TP1 (3))
 - Suivi des connexions Samba sur le serveur
 - En mode texte, suivre les connexions des utilisateurs au serveur Samba, dans les journaux personnels des stations, situés à `/var/log/samba/log.<nom station cliente>`
 - Sur le serveur, la commande **`smbstatus`** permet d'ouvrir une fenêtre d'informations sur l'activité du serveur Samba
Observer au fur et à mesure du travail, le suivi des connexions des clients Windows, avec les indications du nom de l'utilisateur, de la machine cliente et de son numéro IP, la date et l'heure.
Pour imprimer : `smbstatus | lpr`
 - Sous X-KDE, une interface graphique existe pour **`smbstatus`** sous le nom on accède par la commande : *K/Configuration/Informations/Etat de Samba*

TP3 partager des périphériques

Objectifs

Il s'agit de paramétrer les partages de système de fichiers périphériques sur le serveur

Vérifier la visibilité et l'accessibilité aux partages pour les utilisateurs autorisés, le respect des restrictions en écriture

...

Pour l'écriture des sections correspondantes dans `/etc/smb.conf`, se reporter au cours.

1. Partager le lecteur de cdrom
 - Paramétrer ce partage sur le serveur ([Réponse](#))
 - Accéder à la ressource `cdrom` partagée. Que faut-il préalablement effectuer sur le serveur ?
 - Installer les applications `Netscape 4.7` et `Eudora` sur la station, à partir du CDROM.
 - Se connecter par **http** au serveur local Apache de `pcx`, puis à Internet. Bien distinguer les services `samba` et `http` du serveur Linux
 2. Partager éventuellement le lecteur de disquette
 3. Partager un (éventuel) lecteur zip

Supposons que le serveur Linux possède un lecteur ZIP dont le pilote est installé et dont le point de montage est prévu en `/mnt/zip`.
Ecrire la section **[zip]** définissant le partage de cette ressource en lecture pour tous, et en écriture seulement pour `root` ([Réponse](#)).
-

TP4 partager un répertoire public

Mise en oeuvre du partage **[public]**, permettant l'accès complet en lecture/écriture de tous les utilisateurs au répertoire `/home/tmp`.

On pourra juger plus raisonnable de positionner le "sticky bit", comme cela est fait sur `/tmp` (à vérifier !)

1. Créer le répertoire `/home/tmp` et lui accorde les permissions correctes.
 2. Puis paramétrer la section du partage **[public]**
 3. Tester
-

TP5 partager un répertoire réservé à un groupe

Il s'agit de mettre en oeuvre le partage **[stagiaire]** étudié dans le cours. Voici un plan de travail :

1. Créer le groupe `stagiaire` contenant tous les comptes `stagex/stgx`
2. Créer le répertoire `/home/rep-stagiaire`
3. Accorder les droits de propriété de groupe de ce répertoire au groupe `stagiaire`
4. Y positionner les permissions `1770` (pourquoi ?)
5. Créer la section du partage **[stagiaire]**
6. Tester complètement sur la station. Conclusion : ce partage est-il satisfaisant ?

Généralisation

- Tous les répertoires `/home/profdexxx` (`xxx`=code discipline), `/home/classexxx` (`xxx`=code classe)

doivent être partagés en accès complet pour les membres respectifs des groupes `profdexxx` et `classexxx`, par exemple `profde math1, ..., profdetechno1 ..., eleSECA1, ..., elePRE1, ..., eleTERM1, ....`

- Mais ces partages doivent être inaccessibles aux autres groupes. De plus, seul le créateur d'un fichier doit avoir le droit de le supprimer ou le modifier.
- Faire un plan de travail, le réaliser, et tester.
- ** On permet aux profs de créer des sous-répertoires dans `/home/prof`.
Soit `rep-prof1`, un répertoire créé par `prof1`. Comment faire en sorte que les permissions de ce rép. `/home/prof/rep-prof1` et des fichiers qu'il contiendra, soient identiques aux autres fichiers de `/home/prof` ?

TP6 partage d'administration du serveur WEB

- Créer, par exemple à l'aide de **webmin**, un utilisateur **webadmin** (mot de passe=`apache`).
- Paramétrer le nouveau partage **[web]**, du site WEB local c'est-à-dire le répertoire `/home/httpd`
- L'accès complet en écriture, donc la gestion du site WEB par connexion Samba, doit être réservée à **webadmin**. Toutefois, on peut admettre que le groupe **webadmin** (contenant éventuellement d'autres utilisateurs, par exemple `jean`) ait un accès en lecture et personne d'autre !
- Quelles sont les commandes que `root` doit passer pour cela ?
- Sur une station Windows, **webadmin** crée une page d'accueil **accueil.html** pour le site de l'établissement, et la place dans le partage **web**. Vérifier son bon positionnement sur le serveur.
- **webadmin** capture un site sur Internet et le place dans un sous répertoire de /
- `jean` vérifie son accès en lecture aux pages WEB par **protocole Samba**, mais son impossibilité à y écrire, bien qu'il fasse partie du groupe **webadmin** !
- En revanche, **tous** les utilisateurs vérifient leur accès en lecture au site WEB par **protocole http**, en adressant la requête : **http://p0x**

(Pour voir les [réponses](#))

TP7 Linux client de Windows

Monter des partages Windows en lecture seule et en accès complet. Tester à l'aide de **mc**.

Réponses aux questions

TP3 Déclarer la ressource `cdrom` dans `smb.conf` Bien sûr la présence d'un Cd n'est pas suffisante, il doit être monté sur le serveur !

```
[cdrom]
# chemin d'accès au point de montage du CDROM
path = /mnt/cdrom
# accessible à tous les utilisateurs
public = yes
# l'écriture sera forcément interdite
```

writeable = **no**

TP3 Déclarer la ressource zip dans smb.conf

[zip]

```
comment = Partage du lecteur ZIP
path = /mnt/zip
public = yes
write list = root
```

TP6 partage d'administration du serveur WEB

```
root crée cet user
useradd webadmin
passwd webadmin -->apache (ce qui n'est pas un bon choix !)
smbpasswd -a webadmin -->apache
```

puis root ajoute jean dans le groupe webadmin

root accorde à webadmin la propriété de groupe sur l'arborescence /home/httpd (il pourrait se limiter à un sous-rep de /home/httpd/html pour un site précis)

```
chgrp -R webadmin /home/httpd
```

root donne le droit d'écriture w au groupe webadmin sur /home/httpd/html et non sur /home/httpd (il ne faut pas créer d'autres objets à ce niveau)

```
chmod 755 /home/httpd
chmod -R 775 /home/httpd/html
chmod -R 775 /home/httpd/cgi-bin
```

root ajoute dans smb.conf la section :

```
[web]
comment = Gestion du site WEB
path = /home/httpd
valid users = @webadmin
writable = no
write list = webadmin
```

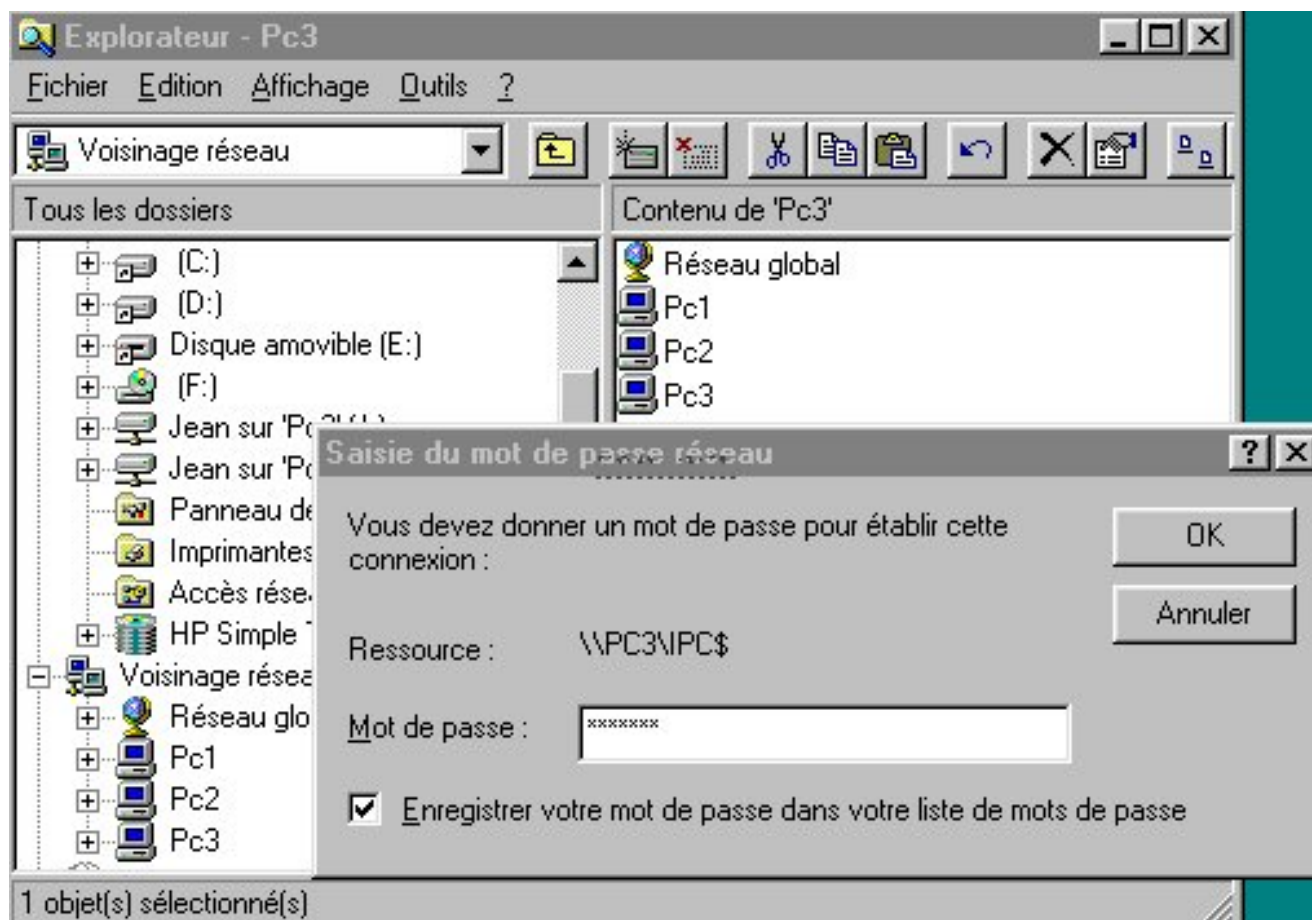
Bien vérifier alors :

- webadmin peut écrire dans W:\html et W:\cgi-bin, mais pas directement dans W:
- jean n'a pas de droit d'écriture sur ce partage, bien que du point de vue permissions Linux, il peut écrire dans /etc/httpd comme membre du groupe webadmin (le vérifier sur le serveur)

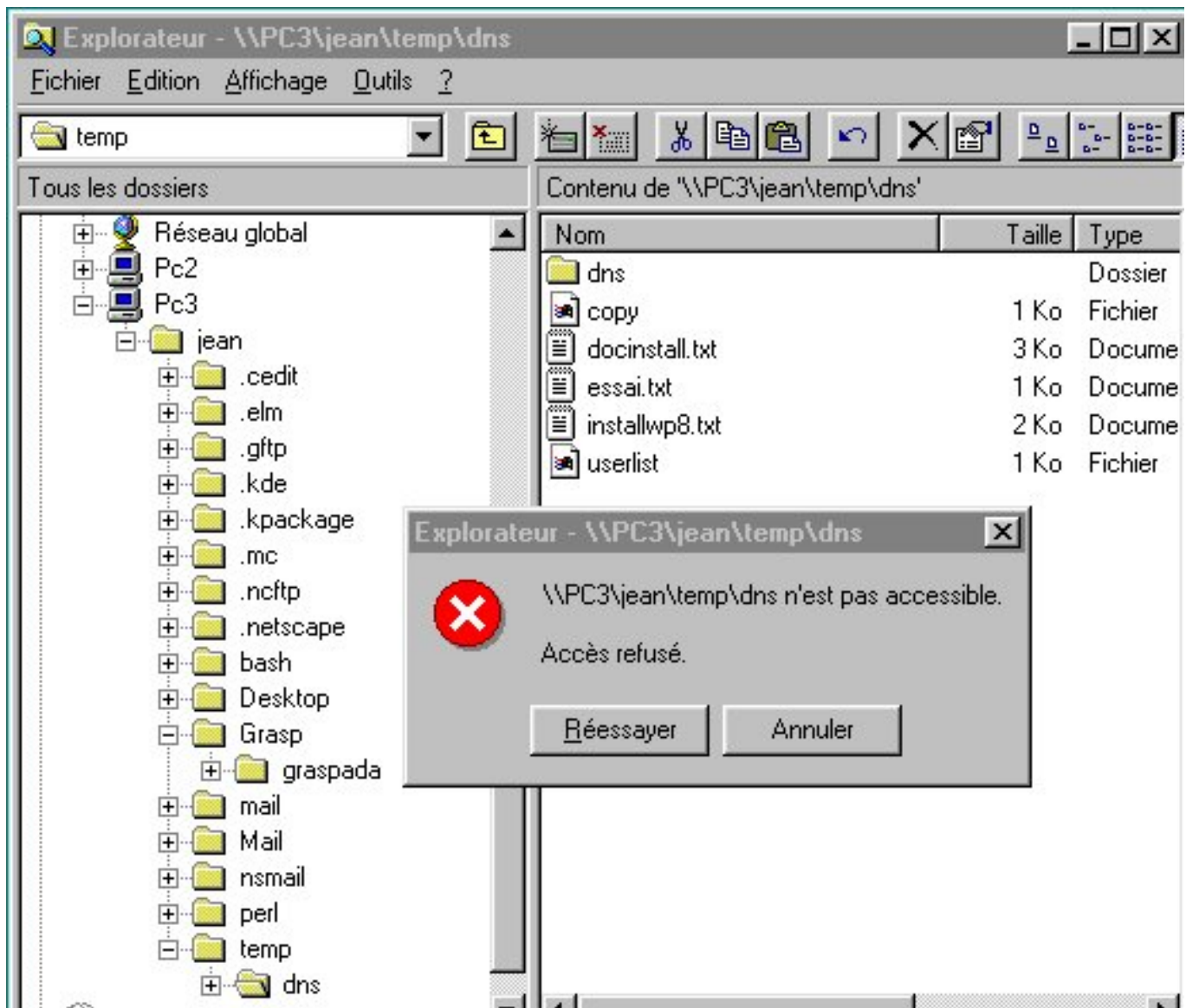


Session de travail sous SAMBA

Si le paramétrage a été correctement effectué, on devrait "voir" dans le *voisinage réseau* le serveur SMB sous le nom de serveur Linux **pxx** et dans le groupe **wcfipen**. Ensuite, il sera possible d'accéder aux ressources avec les droits d'accès définis sur le serveur.

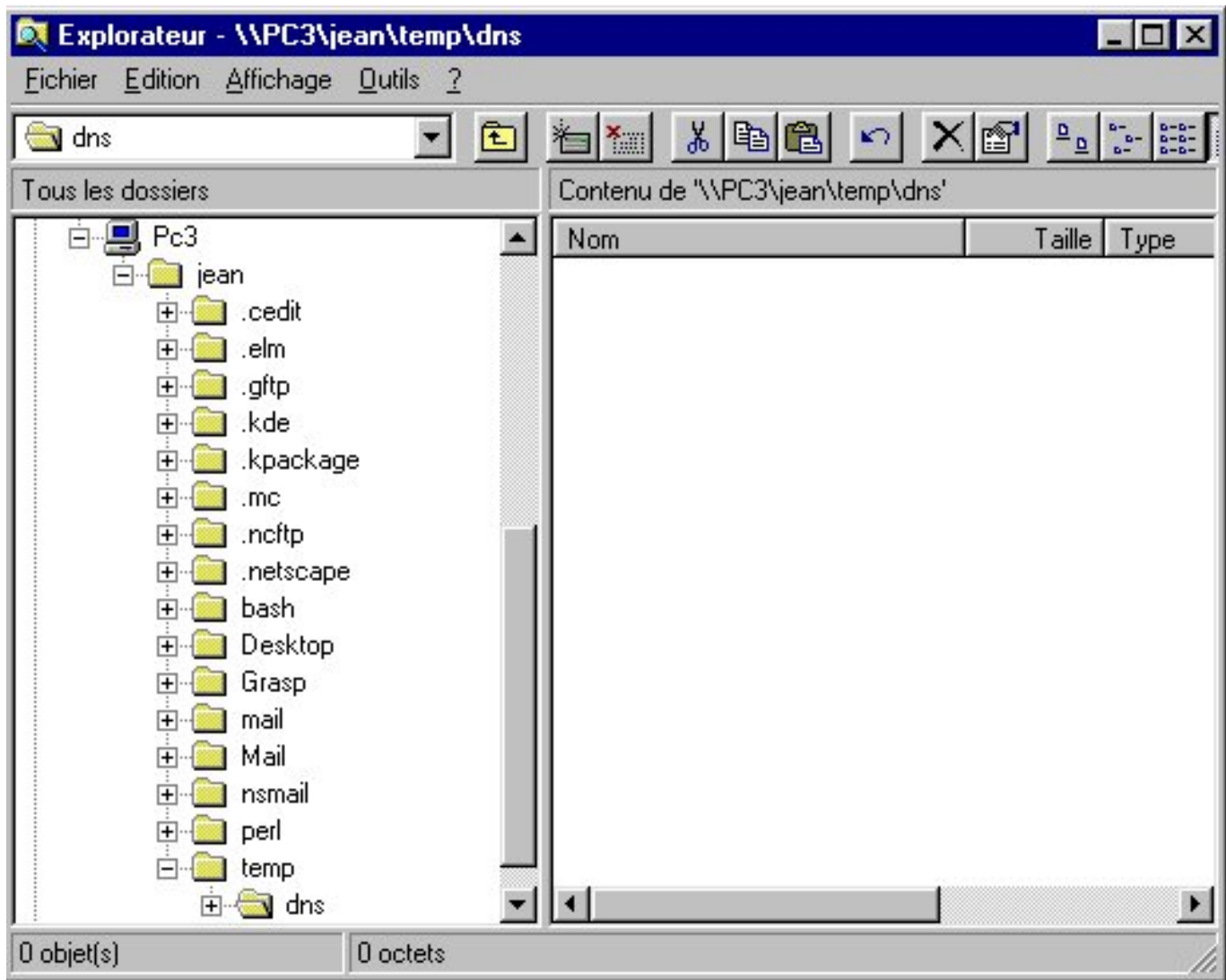


Pour accéder à l'ensemble des ressources partagées sur le serveur SMB, pour l'utilisateur courant, il est demandé le mot de passe du compte, s'il n'a pas été fourni à la connexion réseau du démarrage. Naturellement, ensuite les droits des fichiers Linux s'exercent vis à vis de l'utilisateur connecté. Si certains dossiers pourtant partagés sont inaccessibles, cela est du probablement à l'absence de droits suffisants (notamment le droit x sur un rép).



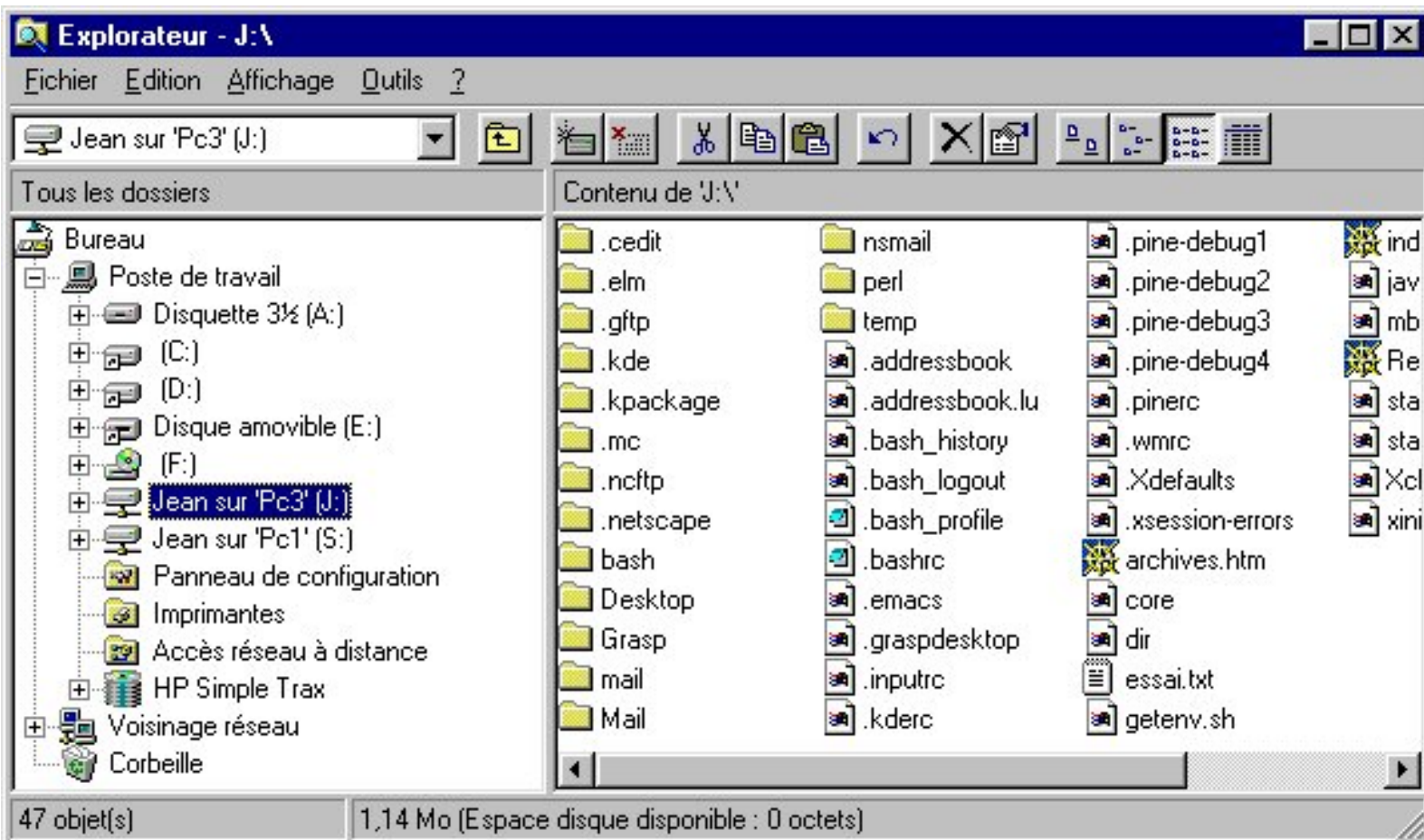
Sur cet exemple, l'utilisateur *jean* installé (pour quelques temps encore) sur la station pc2 sous Windows95 connecté via SMB à pc3, se voit refuser l'accès à un dossier pourtant placé dans son *répertoire personnel*, sur pc3 /home/jean.

Explication : la commande **ls** passée sur pc3 donne : **drw-r----- root root dns/**, ce qui montre que *jean* n'est pas propriétaire de ce répertoire, et plus grave encore, comme user quelconque il n'a aucun droit sur celui-ci ...



Le super-utilisateur `root` a enfin rectifié et lui a accordé le droit de propriété :
`chown -R jean /home/jean/temp/dns.`

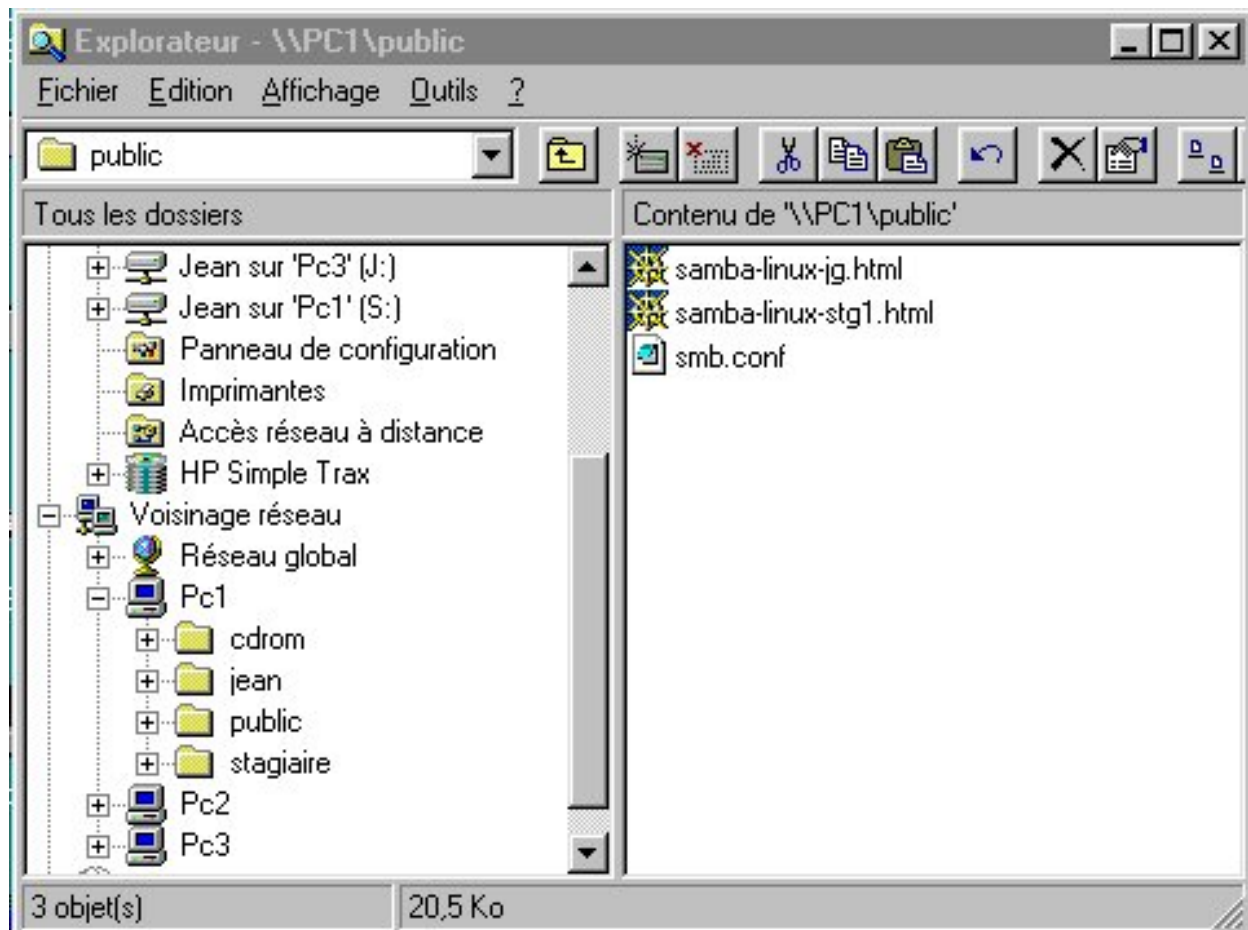
L'accès au répertoire est maintenant permis à *jean*, mais il lui apparaît vide sous Windows !
 Jean, excédé, passe alors sur la console du serveur `pc3`, et lance la commande `cd dns` et ... tout propriétaire qu'il est, il essuie sèchement un refus : *"Permission non accordée"* !
 En revanche, il peut voir son contenu avec `ls dns`, ce qui lui confirme bien qu'il n'est pas vide.
 Pour s'informer de ses droits, il essaie `ls -l dns`, et il reçoit encore un refus !!
 Mais il a (enfin) compris pourquoi et envoie aussitôt un message cinglant à `root`.
 Et vous ? [vérification](#)



Naturellement l'utilisateur gère son espace de travail avec l'Explorateur Windows, avec ses applications comme il le ferait avec un autre type de serveur dédié.

Il peut être pratique de créer une unité de lecteur réseau (ici **J:**) *active au démarrage*.

L'utilisateur se connecte ici à **pc1**, un autre serveur Linux, sur lequel d'autres ressources ont été partagés : Cd-rom, et les dossiers stagiaire (privé) et public (pour tous)





Samba, contrôleur principal de domaine

La résolution des noms NetBios

Le problème

- Le problème essentiel pour permettre la communication entre les réseaux utilisant NetBios et TCP/IP est la "résolution des noms", c'est-à-dire l'utilisation d'un service réseau qui se charge de la traduction nom NetBios de machine / adresses IP
- Il y a 4 procédés de résolution, que le "démon" **nmbd** s'efforce de mettre en oeuvre. Leur ordre d'utilisation est fixé par la clause `name resolve order` dans `/etc/smb.conf`.
Par exemple :
`name resolve order = wins host bcast lmhosts`
- Par défaut, en l'absence de service de résolution de noms (paramétrage standard dans `/etc/smb.conf`), la résolution est tentée par diffusion (*broadcast*), Samba fait alors du "porte à porte".
- Si on utilise la méthode `lmhosts`, le fichier `/etc/lmhosts` doit être renseigné sur chaque machine, comme `/etc/hosts`, sous forme d'une table :
adresse IP (ou nom DNS) nom NetBios
- La commande **nmblookup** **nom** fournit si possible l'adresse IP de la machine, connaissant son nom Netbios.
`$ nmblookup serveur`
querying serveur on 10.177.200.255
10.177.200.100 serveur

Installer un serveur WINS

La meilleure méthode de résolution des noms semble d'activer un serveur **WINS** (= *Windows Internet Name Server*) sur un serveur Samba

Chaque station est affectée à un serveur **Wins**, et sait ainsi auprès de quelle machine elle doit se faire enregistrer, c'est-à-dire faire noter la correspondance entre son nom Netbios et son adresse IP.

Configuration station

Voisinage réseau / Propriétés

propriétés TCP/IP

onglet configuration WINS

Activer la résolution WINS et ajouter l'adresse IP du serveur WINS
et bien sûr redémarrer ..

Configuration d'un serveur Samba

Dans son fichier `/etc/smb.conf`, mettre en premier la méthode de résolution par **wins** et **choisir** une machine Samba dans le sous domaine qui supportera le serveur wins. S'il s'agit de la machine qu'on configure (c'est le plus souvent le seul serveur Samba) on déclare `wins support = yes`, sinon on précise l'adresse IP après `wins server =` .

Comme il ne doit y avoir qu'un seul serveur WINS dans le domaine, on doit n'activer qu'une seule des 2 options :

```
# Activer un serveur Wins pour la résolution des noms NetBios
name resolve order = wins host lmhosts bcast
wins support = yes
# alors ne pas déclarer l'adresse IP d'une autre machine comme étant serveur Wins
; wins server = xxx.xxx.xxx.xxx
```

Samba, contrôleur principal de domaine

- Un *contrôleur principal de domaine (PDC)* est un service chargé du contrôle de l'authentification des requêtes de connexion sur un réseau, par nom de connexion (login) et mot de passe (password).
- Samba peut assurer ce service partiellement (à ce jour) en quelque sorte en émulant les fonctionnalités d'un serveur NT. Il en résulte de meilleures fiabilité et sécurité de la connexion sur le mode client/serveur, à la place d'un pseudo-voisinage réseau et ses aléas ...

Paramétrage du serveur

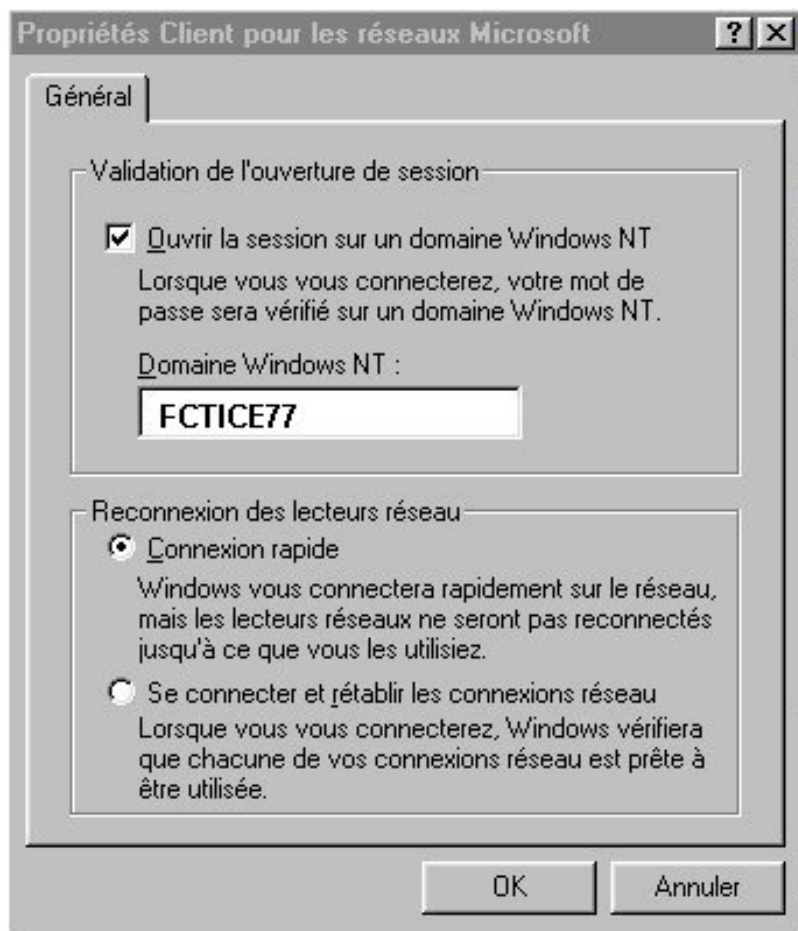
Voici le paramétrage standard dans `/etc/smb.conf`.

```
# dans la section [global]
[global]
workgroup = FCTICE77
netbios name = SERVEUR
server string = Serveur Samba
# active le service PDC
domain logons = yes
# sécurité au niveau utilisateur
security = user
# les mots de passe doivent être encryptés dans le fichier /etc/smbpasswd
encrypt passwords = yes
smb passwd file = /etc/smbpasswd
# pour être aussi serveur de temps
time server = yes
# niveau d'exécution du serveur ?
os level = 34
# ce serveur est le controleur du domaine
domain master =yes
# pour forcer la demande d'authentification pour tout partage (non recommandé)
; revalidate
```

Remarques

- Le nom donné à la rubrique `workgroup` est le nom choisi pour le domaine.
C'est le nom à indiquer sur chaque client Windows (voir ci-dessous).
- La valeur de *netbios name* est le nom du serveur Samba qui l'identifie sur le réseau.
A défaut, le serveur sera visible sur le réseau sous son nom DNS d'hôte (le nom donné sous Linux, lors du paramétrage TCP/IP).
- `domain logons = yes` va activer le service de controleur de domaine.
- l'option `revalidate` renforce la sécurité, mais devient vite fastidieuse pour les usagers

Paramétrage des stations



Sélectionner

1. Voisinage réseau/propriétés
2. Client pour les réseaux Microsoft/Propriétés
3. Cocher *Validation de l'ouverture de session*
4. Ecrire le nom Workgroup, ici **FCTICE77** comme nom de *Domaine Windows NT*
5. Reconnexion des lecteurs réseau : cocher *Connexion rapide*
6. Et bien sûr ... redémarrer !

Résultats

A la prochaine connexion, la boîte de dialogue sera modifiée.

Le 3ème champ *Domaine* y sera ajouté. A chaque demande de connexion, nom et mot de passe sont exigés, leur existence et validité sont vérifiés sur le contrôleur du domaine indiqué, sinon la connexion au réseau est refusée.

Les scripts de connexion

- Principe

Samba accepte la demande de script de connexion Windows (c-à-d les fichiers de commande **.bat**) liés à la connexion d'un utilisateur, et renvoie le fichier correspondant sur la station Windows, afin que celle-ci l'exécute. Ce mécanisme permet donc d'adapter complètement et dynamiquement la configuration du poste client au profil de l'utilisateur.

- Les variables NetBios

On peut utiliser ces variables dans le paramétrage de Samba, mais pas dans les scripts shell

%U	Nom utilisateur
%m	nom NetBios de la machine cliente
%T	date et heure de la [dé]connexion
%I	N° IP de la station
%S	nom du partage courant
%L	nom NetBios du serveur
%G	nom du groupe principal de %U

- Configuration serveur Modifications à apporter à `smb.conf`

[global]

---- voir ci-dessus ----

le script de connexion porte le nom Windows %U de l'utilisateur

```

cette clause suppose l'écriture d'un partage appelé [netlogon]
logon script = %U.bat
# Pour indiquer le chemin du répertoire personnel dans le script
# Ceci va permettre de le connecter avec use net H: /home
logon home = \\%L\%U
.....
[netlogon]
comment = Service de connexion réseau
# répertoire d'accueil choisi pour les scripts de connexion
path = /home/netlogon
# ce partage est privé, invisible et protégé en écriture
public = no
writeable = no
browseable = no

```

- Script pour l'utilisateur stagex

ATTENTION ! Il s'agit d'un fichier "batch" qui va s'exécuter sur la station Windows juste après l'authentification. Il doit donc être écrit avec un éditeur de texte DOS, sur la station, puis ensuite placé sur le serveur dans /etc/netlogon par root ou un utilisateur autorisé.

```

# fichier /home/netlogon/stagex.bat
net use H: /home
net use L: \\serveur\logiciel
net use P: \\serveur\public
net use S: \\serveur\stagiaire
net time \\serveur /set/y

```

- Effets

- Sur le client Windows, après authentification de la requête de connexion, le script attaché à l'utilisateur positionne les lettres des lecteurs réseaux qui "pointent" vers des partages valides et accessibles.
- En particulier le lecteur **H:** désigne bien le répertoire personnel

- Gestion des scripts

- En l'absence d'utilitaire de gestion par interface WEB sur une station cliente, il est indispensable que root délègue à un(e) gestionnaire la maintenance des services samba (mise en place des scripts, installation des applications ...).

Voici comment accorder la gestion des scripts.

- Créer un tel gestionnaire (par exemple, login=admin / passwd=admin / smbpasswd=admin), de groupe primaire admin. Ce groupe doit avoir un accès complet sur /home/netlogon et sur /appli. Mais attention, ce partage doit rester accessible en lecture **et en parcours** pour tous (permissions rx de répertoire) mais pas en écriture, donc avoir pour valeur octale 775

```

chown -R root.admin /home/netlogon
chmod -R 775 /home/netlogon

```

- Droits d'accès au partage [netlogon]
Il faut aussi accorder l'accès en écriture à admin

```

[netlogon]
.....
# partage invisible
browseable = no
# en lecture seule, sauf pour le groupe admin des gestionnaires
writeable=no
write list = @admin

```

- Accessibilité au partage [netlogon]

L'invisibilité du partage (browseable = no) précédent impose de positionner un lecteur réseau pointant vers lui. Pour cela dans le script des gestionnaires, ajouter :

```
# dans /home/netlogon/admin.bat
net use N: \\serveur\netlogon
```

Ainsi seul **admin** pourra accéder aux scripts sur une station cliente, les éditer dans un bloc-notes quelconque, les copier ... bref effectuer ces tâches courantes de gestion.

- Définition de scripts de groupe

- Dans une stratégie de gestion de groupes, il peut être pratique d'utiliser des scripts collectifs. Pour cela, un script **profs.bat** pourrait être appelé par tous les scripts des membres du groupe **profs**, chacun de ces scripts pouvant être ensuite individualisé.
- Soit l'utilisateur **prof1**, membre du groupe général **profs**, devant accéder au partage [maths] de sa discipline (nul n'est parfait ..).
Le gestionnaire **smbadmin** crée le script standard **profs.bat**, puis celui de **prof1**, par exemple :

```
rem script profdemath1.bat
call \\serveur\netlogon\profs.bat
net use M: \\serveur\maths
```

- Avantage évident : s'il est nécessaire de modifier le script de tous les **profs** ...

Exemples d'installations d'applications

- **root** a créé un utilisateur **admin/admin** chargé d'installer les applications sur le serveur, et de la gestion des scripts de connexion.
- Supposons que le partage [logiciels] s'appelle **applications**. Il lui donne la propriété de **/appli**

```
chgrp -R admin /appli
chmod -R 775 /appli
```
- Paramétrage du partage

[logiciel]

```
comment = Applications réseaux
path = /appli
public = yes
writeable = yes
# admin peut installer les applications
write list = admin
```

- **Eudora**

- **admin** crée sur la station le dossier **Eudora** dans **L** :
- il y copie le fichier auto décompactable à partir du Cdrom
- il procède à l'installation dans **L:\Eudora**
- Il crée un raccourci sur le bureau de façon à ce que les paramètres personnels se trouvent dans le répertoire personnel **H**: de l'utilisateur
 - cible : **L:\eudora\eudora.exe** **H:\eudora**
 - démarrer en **L:\eudora**

- root crée le sous répertoire **/etc/skel/eudora**, pour que le dossier **H:\Eudora** soit généré automatiquement à la création d'un nouvel utilisateur

• StarOffice

Naturellement, il s'agit ici d'installer la version Windows sur le serveur Samba.

- Se connecter comme root ou administrateur du partage [logiciel]
- Le lecteur réseau L: pointe vers `\\serveur\appli`
- Clic sur l'archive `so51a_win_33.exe` de 62 Mo ! Indiquer le répertoire de décompression `L:\StarOffice-source`, et Unzip
- Lancer l'installation par Exécuter :
`L:\StarOffice-source\so51inst\office51\Setup.exe /net`
Indiquer `L:\StarOffice` comme répertoire d'installation
- Installation des composants locaux sur chaque station :
Lancer `L:\StarOffice\soffice.exe`, choisir *Installation d'utilisateur standard* et indiquer un répertoire local, par défaut `C:\Office51` est indiqué.
- L'installation a créé une entrée dans le Menu Démarrer du disque de la station, dans `C:\Windows\Menu Démarrer\programmes`, qui pointe vers `L:\StarOffice\soffice.exe`. Si on veut que tous les utilisateurs en bénéficient, il suffit de le recopier dans tous les répertoires `/home/user/Démarrer/Programmes`, bien sûr avec un petit script pour automatiser ...
- Lors du premier lancement sur une nouvelle station, le processus d'installation local sera déclenché.



Il s'agit de mettre en pratique les notions acquises dans ce chapitre, par étapes, à partir d'une configuration standard de Samba.

Pour cela, il est recommandé de partir de la configuration `smb.conf` obtenue à l'issue du [TP précédent](#).

1. Situation de départ

Le paramétrage standard de `smb.conf` doit permettre à un utilisateur `stagex` de se connecter au serveur situé dans le même `workgroup`, par exemple `fctice77x`, et d'accéder à son répertoire personnel, au partage WEB et à divers répertoires partagés.

2. 1ère étape : installer les services WINS et Contrôleur de domaine

- Donner le nom NetBios *serveurx* au serveur Samba, et constater que le serveur apparaît maintenant sur les stations sous ce nom.
- Activer le service **WINS** sur le serveur et le client
- Activer le service PDC sur le serveur : quand on relance samba, le message suivant est ajouté au fichier `/var/log/samba/log.nmb` "Samba is now a logon server for workgroup `fctice77x` on subnet `10.177.200.(100+x)`"
- Se connecter sous le compte `stagex`. Observations : bureau, voisinage réseau ?

3. 2ème étape : les scripts de connexion

- root crée le partage [netlogon] qui pointe vers `/home/netlogon`, et accorde à admin l'accès pour y gérer les scripts.
- Puis admin crée les scripts pour les divers utilisateurs, à l'aide de Wordpad par exemple, sur une station Windows en décidant d'une affectation de lecteurs réseaux, telle que :
fichiers `/home/netlogon/xxx.bat`
net use H: /home

```
net use L: \\serveur\logiciel
net use P: \\serveur\public
net use N: \\serveur\netlogon
net use W: \\serveur\web
net use S: \\serveur\prof
net time \\serveur /set/y
```

- Les différents utilisateurs se connectent sur le domaine et testent leur accessibilité aux différents partages (parcours, lecture et écriture). Est-ce satisfaisant ?

4. 3ème étape : les profils

- Modifier la base de registre des stations pour que le bureau et le menu démarrer soient chargés à partir du répertoire personnel à chaque connexion.
- Etudier une procédure d'automatisation de ces différents paramétrages lors de la création d'un nouvel utilisateur.

Annexes

Lecteur réseau vers un serveur SAMBA

- Dans Windows on peut créer des raccourcis vers des ressources réseaux, principalement des partages de répertoires. Associés à des lettres, ils portent le nom de lecteurs (ou unités) réseau.
Par exemple la lettre K : peut être associée au chemin \\Serveur\homes
- Cette notation obéit à la syntaxe UNC (Universal Naming Convention) **\\machine-réseau\répertoire**
- Par exemple \\Serveur\annales, où **Serveur** est le nom NetBios du serveur Samba, et **annales** le nom donné à un partage au sens Samba.
C'est-à-dire le nom donné à la section [annales] du fichier /etc/smb.conf, et non pas le nom du répertoire correspondant, indiqué par le paramètre chemin, path = .., présent dans la description de ce partage.
- On dit souvent plus simplement que F : pointe vers la ressource (ou le partage) \\Serveur\annales. La situation réelle du répertoire annales sur le serveur est ainsi masquée à l'utilisateur de la station Windows. Par contre le détail de l'arborescence dont la racine est annales, vue d'une station Windows, est identique à l'arborescence réelle sur le serveur Linux (à part le remplacement des / par des \)



Mise en place d'une messagerie locale

L'intérêt d'une messagerie locale à un établissement dépasse le simple échange de messages en format texte. C'est certainement le moyen le plus simple de transmettre des fichiers de toute nature, en les acheminant attachés aux messages. Ce service est en mesure d'éviter la lourdeur de l'installation de partages. Ceux-ci doivent être réservés aux échanges importants et durables entre membres de groupes stables.

Paramétrage du serveur de courrier

Installation du serveur

- Installation

Voici pour une distribution Mandrake, les paquetages à installer ou à mettre à jour.

Le cas échéant, pour plus de facilité, on peut utiliser l'outil graphique KPackage.

```
mount /mnt/cdrom
```

```
cd /mnt/cdrom/Mandrake/RPMS
```

pour installer le serveur de courrier entrant (POP3) :

```
rpm -ivh imap-4.5-5mdk.i586.rpm
```

pour installer le serveur de courrier sortant (sendmail)

```
rpm -ivh sendmail-8.9.3-11mdk.i586.rpm
```

```
cd
```

```
umount /mnt/cdrom
```

- Le (fameux) fichier de config de sendmail

Renommé pour sa complexité, ce fichier **/etc/sendmail.cf** ! A tel point qu'il est explicitement recommandé dans sa documentation de ne pas le modifier "à la main", mais d'utiliser **Linuxconf** pour une éventuelle retouche !

- Sur un petit réseau en l'absence de DNS, les adresses e-mail sont composées directement avec le nom du serveur (ici p00) gérant à la fois smtp et pop3

Avec Linuxconf

Réseau/système de messagerie (sendmail)

Configurer information de base

Présenter votre système comme --> [x] Gérer le domaine fctice77.fr

Serveur de messagerie --> p00.fctice77.fr

Fonctionnalité DNS --> [x] ne pas utiliser le DNS

Accepter la "régénération" par linuxconf et activer les changements, ce qui provoque :

```
/etc/rc.d/rc3.d/S80sendmail restart
```

- Pour vérifier que sendmail "tourne" :

```
ps aux|grep sendmail
```

ou directement avec

```
/etc/rc.d/init.d/sendmail status
```

- Rappel : l'utilitaire **ntsysv** permet de lancer automatiquement sendmail au démarrage.

Paramétrage des boîtes de messagerie personnelle

Nous verrons qu'il est plus simple de placer les boîtes de courrier des utilisateurs dans leur répertoire personnel. Pour chaque utilisateur, on créera un sous-répertoire `/home/$USER/messagerie` qui lui appartient avec les droits 700

Installation du logiciel Eudora

- **admin** est le gestionnaire chargée de l'installation des applications. Pour cela il possède un droit d'écriture sur le partage **logiciel** c'est-à-dire sur **/appli**
- **admin** crée sur la station le dossier **eudora** dans **L :**, il y copie le fichier auto décompressable à partir du Cdrom, puis il procède à son installation dans **L :\Eudora**
- Il crée un raccourci sur le bureau de façon à ce que les paramètres personnels se trouvent dans le répertoire personnel **H :** de l'utilisateur
 - cible : **L :\eudora\eudora.exe** **H :\messagerie**
 - démarrer en **L :\eudora**
- **root** crée le sous répertoire **/etc/skel/messagerie**, pour que le dossier **H :\messagerie** soit généré automatiquement à la création d'un nouvel utilisateur

Paramétrage des stations Windows

- Tout utilisateur, par exemple **jean**, titulaire d'un compte sur le serveur peut paramétrer son logiciel de messagerie habituel sur un poste Windows pour une boîte de messagerie locale.
- Normalement le lancement d'Eudora sur la station provoque la demande de paramétrage de la boîte de l'utilisateur.
Voir pour l'installation réseau de [Eudora](#)
- Attention !
Le nom du serveur et son numéro IP doivent être connus sur les machines clientes. En l'absence de DNS, bien vérifier dans le fichier **C :\windows\hosts** la présence d'une ligne renseignant sur le serveur comme :
10.177.200.100 p00.fctice77.fr p00
- Paramétrage habituel (sur Eudora)
compte e-mail pop3 : **jean@p00** (ou **jean@p00.fctice77.fr**)
adresse de retour : **jean@p00**
serveur smtp : **p00**
- Comme on reste connecté sur un réseau local, on peut régler le logiciel client de messagerie pour qu'il consulte régulièrement la boîte (sur Eudora *Tools/options/Checking mail/check for mail every 10 minutes*)
- Sur le serveur, les messages sont stockés temporairement dans des fichiers **/var/spool/mail** portant le nom des comptes, puis sauvegardés dans **~/mbox**. On constate notamment que le code des pièces jointes est inclus dans les messages eux-mêmes.
- On pourrait très bien gérer son courrier sur une station Linux, avec **mail** en mode console ou un client graphique comme **Kmail** et s'en servir pour envoyer et recevoir de toutes les autres stations.



1. Tester l'utilisation de **p00** comme unique serveur de messagerie sur l'ensemble des stations Windows (avec Eudora) et Linux (avec Kmail)
Automatiser la réception du courrier toutes les 5 minutes
Observer le stockage temporaire du courrier sur le serveur **p00**
Où sont stockés les fichiers joints ?

2. Ensuite, mettre en oeuvre un service de messagerie dans chaque groupe de travail fictice77x
Echanger du courrier entre le serveur en ligne de commandes avec **mail**, et les stations.

Aller vers une installation automatisée

- L'exécutable se trouve sur le serveur à `/appli/eudora/eudora.exe` et est accessible sur les stations sous le nom `L:\Eudora\eudora.exe`
- Sur chaque station le raccourci suivant a été installé
 - cible : `L:\Eudora\eudora.exe` H:\Eudora
 - démarrer en `L:\Eudora`
 - nom : *Messagerie interne*
- Créer un répertoire vide **/eudora** dans **/etc/skel**, afin qu'à toute création d'un nouvel utilisateur, un sous-répertoire **eudora** soit créé dans son répertoire personnel. Y placer un fichier **eudora.ini** déjà préconfiguré avec des choix judicieux d'options par défaut, de façon à ce que l'utilisateur n'ait que son nom de connexion à ajouter.
- Grâce au raccourci précédent pointant vers **H:\Eudora**, le fichier `eudora.ini` modifié contenant les paramètres personnels de l'utilisateur y seront déposés dans **H:\Eudora**, lors de sa première connexion au serveur de messagerie
- Avantages de ce dispositif
 - L'installation du client de messagerie (ici Eudora) est effectuée dans le répertoire **/appli/eudora** protégé en écriture.
 - Les mises à jour sont simplifiées !
 - Le même raccourci sur le bureau est utilisé par TOUS, puisqu'il pointe automatiquement vers le répertoire personnel de l'utilisateur, assurant ainsi la confidentialité du courrier.
- Ceci pourrait être amélioré, en écrivant un script, exécuté à chaque création d'un nouvel utilisateur, qui générerait le fichier de paramétrage `eudora.ini` adapté à cet utilisateur, à partir d'un modèle initial.



Scripts de messagerie

1. Exemple de script d'envoi collectif

```
#!/bin/bash
# script message0.sh
for nom in stage*
do
mail $nom@p00 <<EOF
bonjour a tous
ceci est le premier essai d'envoi
de message par un script shell,
en utilisant mail sur p00
A +
Le "root"
EOF
done
```

Observer le stockage temporaire des messages dans les fichiers `/var/spool/mail/$nom`.

Quand le message a été lu par un destinataire, une copie est stockée dans son répertoire personnel : `$HOME/mbox`

2. Script `message1.sh`

Envoi du message contenu dans un fichier texte dont le nom est passé en 1er paramètre à chaque utilisateur de la liste qui suit.

Exemple d'appel

```
./message1.sh attention.txt jean toto
```

3. Script `message2.sh`

Envoi d'un message collectif à tous les utilisateurs dont le nom commence par exemple, par "stage"

```
for nom in $(cat /etc/passwd | grep "^stage" | cut -d: -f1)
```

4. Script `message3.sh`

Envoi d'un message général à tous les utilisateurs actuellement connectés

```
for nom in $(who | cut -d" " -f1)
```

[Proposition de corrigés](#)

Proposition de corrigés

● Script `message1.sh`

```
#!/bin/bash
```

```
# appel : message1.sh fichier liste_users
```

```
# Le fichier texte est envoyé à chacun des utilisateurs de la liste
```

```
[ $# -lt 2 ] && (echo "Syntaxe $0 texte-message liste-utilisateurs";exit 1)
```

```
# Attention: les ( ) sont obligatoires car && est plus prioritaire que ;
```

```
texte=$1
```

```
shift
```

```
for nom in $@
```

```
do
```

```
# variante mail $nom@p00 < $texte
```

```
cat $texte | mail $nom@p00
```

```
done
```

● Script `message2.sh`

```
#!/bin/bash
```

```
# envoi à tous les users stagex
```

```
for nom in $(cat /etc/passwd | grep "^stage" | cut -d: -f1)
```

```
do
```

```
mail $nom@p00 <<FIN
```

```
Bonjour à tous,
```

```
.....
```

```
A +
```

```
Le "root"
```

```
FIN
```

```
done
```

● Script `message3.sh`

```
#!/bin/bash
```

```
# envoi à tous les utilisateurs connectés
```

```
echo "message envoyé le $(date) a " >> utilisateurs.send
```

```
for nom in $(who | cut -d" " -f1)
do
mail $nom@p00 <<EOF
bonjour a tous
Attention ! le root vous parle !
Déconnexion très tôt aujourd'hui ...
à 18h, pour travaux de maintenance réseau
Bonne soirée quand même ;- )
A +
Le "root" de p00
EOF
echo "$nom " >> utilisateurs.sent
done
```



Exploration de SambaEdu

Présentation

- Site de SambaEdu à <http://www.linux-france.org/prj/edu/smbaclg/>
- Source téléchargeable à : <ftp://ftp.linux-france.org/pub/prj/edu/smbaclg>
- Liste de diffusion : envoyer le message *subscribe* à majodormo@etab.ac-caen.fr
- Auteur : Olivier Lecluse olivier.lecluse@linux-france.org

L'objectif est de fournir une configuration "clé en mains" pour un petit réseau d'établissement. Le kit standard contient un ensemble de fichiers scripts permettant notamment :

- d'automatiser la gestion des comptes
- de sécuriser le partage des répertoires
- d'installer les applications sur le serveur.
- de permettre la gestion aisée d'un intranet : outil de capture de sites, autorisation de sortie "Internet" en fonction des utilisateurs.

L'accès aux comptes et aux services sur le serveur LINUX s'effectuent par le protocole Samba. Le compte **admin** permet d'administrer complètement à distance les partages, les comptes utilisateurs, les groupes, la capture de sites, ... à travers une interface WEB dialoguant via une passerelle CGI, avec des scripts BASH.

Compléments

● **Installation du "Power-kit"**

Un complément de fonctionnalités (le "Power-kit") permet de bénéficier de services supplémentaires :

- une messagerie interne
- un serveur de noms (DNS) pour le réseau local
- un serveur d'adresses dynamiques ip (DHCP)

● **Créer les comptes à partir de GEP**

Voir la doc correspondante

Installation

1. Préparation

- ouvrir une session **root**
- **mount /dev/fd0** pour monter une disquette contenant le paquetage **latest.tar.gz**
- **cp /mnt/floppy/latest.tar.gz /root/tmp** on peut utiliser **mc**
- **cd /root/tmp**
- **tar -xzf latest.tar.gz** décompresse (z), extrait (x), le fichier (f) indiqué.

2. Résultats de cette 1ère étape

- création de 3 répertoires dans **root+/tmp** :
v2.03/ , **pkitt1.02/** (droits 750)
gep2smb/ (755)
- le répertoire **v2.01** contient le script principal d'installation qui va décompresser l'archive **sambaedu_2.03.tar.gz**

3. Création des comptes initiaux et des partages

- **cd /tmp/v2.03**
- **./install.sh** lance l'exécution du script d'installation
- arrêt des serveurs Samba et Apache
- demande du mot de passe de admin --> *admin*
- mot de passe de eleve --> *eleve*
- mot de passe de prof --> *prof*
- adresse ip du serveur --> *10.194.2.100+x*
- nom du domaine --> (*WCFIPENx*) il s'agit du nom *workgroup*

4. Résultats obtenus

Le processus d'installation peut être suivi et compris en lisant le script **install.sh**
 On y voit notamment :

- la copie de **sambaedu_2.03.tar.gz** directement dans la racine / , puis sa décompression
- la création de 3 groupes : samba (GID=5000), eleves(5001), profs(5002)
- la création de 3 comptes : admin (UID=5000), eleve(5001), prof(5002) de groupe primaire samba.

Les rép. personnels sont tous installés dans **/serveur/home/**

5. Examens des rép. et fichiers modifiés

- L'examen de **linuxconf** ou directement des fichiers **/etc/passwd** et **/etc/group** montrent les appartenances de groupe.

Le groupe **samba** est défini comme le *groupe primaire* de tous.

- **samba** = {admin, prof, eleve}
- **eleves** = {eleve}

- `profs = {prof, admin }`

○ Dans le rép. du serveur Apache on observe :

- Le répertoire `/home/httpd/bin` est créé et contient des exécutables : `crypt`, `exe ..`
- Le répertoire `/home/httpd/html/admincgi` est créé et contient des fichiers `html`.
Ceux-ci seront utilisés par `admin`, pour gérer le réseau à distance, sur une station Windows, en dialoguant avec le serveur WEB Apache.
- Le répertoire `/home/httpd/cgi-bin` contient les scripts d'administration, appelés et exécutés par les formulaires de l'interface d'administration.

6. Examen de la configuration Samba mise en place

La consultation directe de `/etc/smb.conf` ou par `linuxconf` montre :

- le nom du workgroup est bien `WCFIPENX`
- le serveur a été doté d'un nom netbios, `SERVEUR`, distinct de son nom Linux, sous lequel il sera reconnu des stations.
- la section définissant le rep. perso. prédéfini `[homes]` a été supprimée.
A la place les répertoires personnels sont définis par la section :

`[home]`

```
comment = Repertoire prive de %U sur %h
path = /serveur/home/%U
write list = admin,%U
```

`%U` est le nom de login de l'utilisateur, `%h` est le nom du serveur Linux (=HOSTNAME)

- Les mots de passe SMB sont cryptés, et synchronisés avec les mots de passe Linux.

7. Complément : installer le "power-kit"

Il faut éventuellement installer les serveurs suivants : `Web Apache`, le serveur de noms `Bind`, le serveur de messagerie `Sendmail` et `POP3`, et le `DHCP`.

Pour savoir quels paquetages sont déjà installés, lancer l'utilitaire **kpackage** sous X, puis *fichier/Chercher un package* Pour installer un nouveau paquetage, voir [la commande rpm](#)

Pour installer, commande `./installpk.sh` dans le répertoire `pk1.02`

Installation des stations Windows9x

Paramétrage du voisinage réseau (sans service DHCP)

- onglet *Configuration* : les composants réseaux à installer sont
 - l'adaptateur réseau qui doit être lié au protocole TCP/IP
 - le client pour les réseaux Microsoft
 - le protocole TCP/IP
- onglet *Identification* : le groupe de travail doit être le nom de domaine `workgroup` déclaré à l'installation.
- onglet *Contrôle d'accès* : au niveau ressources

- Propriétés du client Microsoft
 - Cocher *ouvrir la session sur un **domaine Windows NT***
 - Comme *Domaine Windows NT*, écrire le nom de domaine Workgroup
- Propriétés TCP/IP
 - Indiquer l'adresse IP de la station et le masque de sous-réseau
 - NetBIOS doit être activé avec TCP/IP
 - Passerelle : indiquer l'adresse IP du routeur (pour connexion Internet)
 - Configuration DNS : indiquer le domaine et l'adresse IP de la machine DNS (serveur de noms) du fournisseur d'accès Internet (par exemple `ac-creteil.fr` et `195.98.246.50`)
- Le cryptage des mots de passe est activé. Il faut utiliser l'utilitaire `crypt.reg` sur les clients 95

Tester !

1. Utilisateurs

Se loguer tour à tour comme eleve, prof

Vérifier pour chacun, l'accessibilité et les droits sur les partages :

- [home] accès au rép. perso, unité réseau K:
- [public] P:
- [profs] J:
- [install] contient les fichiers de config .reg des stations, I:
- [logiciel] logiciels accessibles à tous, L:

Constater la cohérence avec les déclarations du fichier `smb.conf`

et l'impossibilité d'effectuer une tâche d'administration à distance, comme sur la console du serveur !

2. Administrateur

- Tester les outils d'administration en se connectant au serveur WEB comme `admin` à `http://adr-ip/admincgi/index.html`, en fait requête au fichier situé sur le serveur à `/home/httpd/html/admincgi/index.html`
- La page *Configuration générale* propose l'accès au service SWAT, au port 901
S'il s'agit de configurer Samba, donc d'agir sur `/etc/smb.conf`, il faut entrer dans `swat` comme `root`.
- Créer de nouveaux utilisateurs :
 - Passer à la page *Gestion des utilisateurs* (`users.html`)
 - puis à `/cgi-bin/createusrhtml.cgi` *Création d'un utilisateur*
 - Le formulaire demande le nom de login, le mot de passe, son groupe primaire et un utilisateur modèle qui définit le profil.

- La validation du formulaire provoque l'exécution du script `/cgi-bin/createusr.cgi`
- par ex. `toto`, avec mot de passe `zig` dans le groupe des élèves, avec comme utilisateur modèle `eleves`
`dupont`, avec mot de passe `dup` dans le groupe des profs, avec comme utilisateur modèle `profs`
Vérifier la répercussion sur le serveur :
 - contenu du répertoire personnel `/serveur/home/toto`
 - appartenance de `toto` au groupe `eleves`
- Tester l'utilitaire de capture de sites (utilisant la commande linux **wget**)
 - Attention, la commande **wget** n'est pas installée par défaut.
Monter le cd-rom et se placer dans `/mnt/cdrom/Mandrake/RPMS` (pour une telle distribution)
Passer la commande `rpm -Uvh wget-1.5.3...rpm` (ou installer en mode graphique)
 - Sur une station Windows cliente, la capture doit être effectuée sous le compte `admin`,
Par l'interface WEB, émettre la requête `http://10.194.2.10x/admincgi/capture.html`
 -  Exemple
Soit à capturer, par exemple, une partie du site de la CNIL à `http://www.cnil.fr`
Commande *Capturer un nouveau site*, préciser l'URL du site à capturer et le sous-rép. de `/serveur/admin/web/cnil` (ie `K:\web`).
 - la capture peut s'interrompre à tout moment, et est consultable hors connexion dans `K:\web\cnil`, qui en fait est un lien qui pointe vers `/home/httpd/html/cnil/`.
C'est ce qui permet à tous de le consulter.
-  Exercice
Capturer le site de SambaEdu à l'URL `http://www.linux-france.org/prj/edu/sambac1g/`



Introduction au shell Bash

Le shell

Un interpréteur de commandes (le "shell", la coquille qui entoure le "noyau" du système) est un programme qui sert d'intermédiaire entre l'utilisateur et le système d'exploitation.

Sa tâche essentielle est l'exécution de programmes.

Pour cela, il effectue (en boucle infinie) :

- la lecture d'une ligne
- sa compréhension comme une demande d'exécution d'un programme avec d'éventuels paramètres.
- le lancement de ce programme avec passage des paramètres
- d'éventuelles redirections d'entrées-sorties
- les exécutions de scripts (fichiers de commandes)

Démarrage du shell

- Lors de la création de son compte, un utilisateur est associé à un type de **shell**
- Lire le fichier **/etc/passwd** : le dernier champ contient le nom du fichier exécutable (shell par défaut) **/bin/bash**
- Le shell associé est ainsi lancé automatiquement dès la saisie du login utilisateur.
- Il poursuit sa configuration en exécutant des scripts globaux à tous les utilisateurs et des scripts liés au compte et qui permettent une personnalisation.
- Enfin, il affiche le prompt et se met en attente de la lecture d'une commande.
- Jusqu'à la commande **exit**, pour quitter le shell (ce qui équivaut à se déconnecter (logout))

Les scripts de connexion

1. d'abord le script **/etc/profile** communs à tous les users y compris **root**. On y trouve notamment la définition de **umask**
2. celui-ci cherche à exécuter tous les scripts **/etc/profile.d/*.sh** (parcourir **alias.sh** et **numlock.sh**)
3. puis il y a exécution de **\$HOME/.bash_profile** (la variable **\$HOME** contient le chemin vers le répertoire personnel). Il s'agit ainsi d'un fichier de démarrage personnel et paramétrable.
4. A son tour il exécute **\$HOME/.bashrc** dans lequel il est recommandé de placer toutes les fonctions ou alias personnels (car **.bashrc** est exécuté dans tout shell)
5. Enfin le précédent script exécute **/etc/bashrc**, dans lequel on place les alias globaux et la définition symbolique du prompt **\$PS1**
6. Puis le prompt utilisateur s'affiche et le shell attend une commande ...

Personnalisation des commandes bash

- **/etc/bashrc** étant le dernier script d'initialisation du shell bash, **root** peut y définir des alias globaux pour tous les utilisateurs
- Exemple avec **vi** (pour utiliser l'éditeur de Midnight Commander lancer **mc**)

```
# vi /etc/bashrc
alias ll="ll | less"
alias x="startx"
alias m="mc"
:wq (pour écrire dans le fichier et quitter vi)
```
- Puis se reloguer (exit) pour que ces nouvelles commandes soient prises en compte par le nouveau shell.

Personnalisation du login utilisateur

Chaque utilisateur peut ajouter des commandes shell au fichier de profil personnel, `~/.bash_profile`

Par exemple, voici ce que j'ai mis à la fin de ce fichier :

```
clear
salut="Bonjour $USER !\nJe te souhaite bon courage ...\n\
# le dernier \ pour pouvoir continuer la commande sur la ligne suivante
# $( ..) pour obtenir le résultat de l'exécution de la commande incluse
Nous sommes le $(date) "
# -e option indispensable pour interpréter les \n
echo -e $salut
```

Les variables d'environnement système

- La liste en est accessible par la commande : **env**
- La commande **echo** permet d'obtenir la valeur d'une telle variable.
Par exemple : **echo \$PATH**, **echo \$USER**
- Ajout d'un nouveau chemin : attention à ne pas écraser la liste des chemins existants (**PATH** en majuscules !)
 - **PATH="\$PATH:/home/jean/bin"**
pour ajouter le chemin vers les exécutables du rép. personnel (Attention ! pas d'espace autour du symbole =)
 - **PATH="\$PATH :./"**
pour toujours ajouter le répertoire courant (non présent par défaut)
- La variable **\$HOME** contient le chemin du rép. personnel.
La commande `cd $HOME` est abrégée en `cd`
- La variable **\$USER** contient le nom de l'utilisateur
- **\$SHLVL** donne le niveau du shell courant

Facilités de saisie des commandes

Comme les commandes Unix sont souvent longues à saisir, diverses facilités sont offertes :

Historique des commandes

Cette liste numérotée est accessible en tapant **history** | **less** Pour relancer la commande numéro **n**, saisir (sans espace) **!n**
On peut aussi parcourir les précédentes lignes de commandes avec les flèches (comme *doskey*) et les éditer. Ceci permet très facilement de reprendre une précédente commande pour l'éditer et la modifier.

Le clic-droit

Dans un terminal console, sélectionner un texte quelconque. Un clic-droit recopie ce texte sur la ligne de commande, même dans une autre console.

L'opérateur tilde

Le caractère tilde `~` (alt 126) **seul** renvoie au **rép. personnel** de l'utilisateur actuel.

Si l'utilisateur actif est toto, chaque occurrence du caractère `~` est remplacé par le chemin `/home/toto`

Le tilde `~` suivi d'un nom d'utilisateur, par ex `jean`, renvoie au **rép. personnel** de `jean`, c-à-d `/home/jean`

Ainsi par cette commande **cd ~stagiaire3** tente en vain d'aller dans le rép. `/home/stagiaire3`



TP 1

1. Personnaliser le script **.bash_logout** situé dans votre répertoire personnel pour que le contenu du cache de Netscape soit effacé au moment de votre déconnexion
Puis comme **root**, compléter le script "modèle" `/etc/skel/.bash_logout`, afin que ce "nettoyage" soit effectué pour tout nouvel utilisateur.

- Expérimenter les "tab" et "clic-droit".
Saisir `echo -n "Bonjour $USER ! Nous sommes le "; date`
Puis utiliser le "clic-droit" pour exécuter cette commande dans une autre console.
- [root@pc5 /root] **cat** > **essai.txt** <Entr> *envoie la saisie dans le fichier*
Ceci est un essai tout simple ! <Entr>
mais tout-à-fait intéressant. <Entr>
<CTRL-D> *caractère fin de fichier sous Linux*
[root@pc5 /root] **cat** **essai.txt** *envoie le contenu à la console*
[root@pc5 /root] `cp` **essai.txt** `~jean` *copie le fichier dans /home/jean*

Compléter une commande

Lorsqu'on tape une commande en ligne la touche **TAB**, l'interpréteur cherche à compléter le nom du fichier.
`home/toto ]$ less /etc/fs TAB`

S'il y a plusieurs propositions, il y a attente d'un complément d'info de la part de l'utilisateur (avec un "tut").
 Un autre **TAB** et l'interpréteur affiche toutes les possibilités ou en indique le nombre, s'il y en a beaucoup !



TP 2

```
$ cd /etc <TAB>
there are 111 possibilities. Do you really wish to see them all ? (y or n)
$ cd /etc/s <TAB>
security services smb.conf syslog.conf etc ..
$ cd /etc/sys<TAB>      # on tape y, le système complète s et ... attend
sysconfig syslog.conf syslog.conf.inn
$ cd /etc/sysc<TAB>    # on ajoute c, le système complète aussitôt à sysconfig
$ cd /etc/sysconfig/ <Entr>
```

Exo : poursuivre ainsi jusqu'à afficher le contenu du fichier de configuration de l'interface Ethernet
`/etc/sysconfig/network-scripts/ifcfg-eth0`

Désigner un ensemble de fichiers

- Travailler avec le shell nécessite souvent de manipuler des ensembles de fichiers. L'utilisation de caractères spéciaux (appelés aussi **méta-caractères**) dans les noms de fichiers, permet de générer des modèles pour désigner ces ensembles. Il existe quatre constructeurs de modèles *****, **?**, **[]** et **^**.

Modèle	Signification
*	remplace une chaîne de longueur qcq, même vide
?	remplace un seul caractère qcq
[]	un caractère qcq de la liste ou de l'intervalle
[^]	n'importe quel caractère sauf ceux de la liste

Attention ! en raison de certaines ressemblances, ne pas confondre ces constructeurs d'ensembles de fichiers avec les expressions rationnelles (utilisées par exemple dans [grep](#) ou [sed](#))

- Un modèle de la forme **X*Y** où X et Y sont 2 chaînes quelconques, éventuellement vides, désigne l'ensemble des noms de fichiers de la forme XZY où Z est une chaîne quelconque elle aussi éventuellement vide.
- Un modèle de la forme **X?Y** désigne l'ensemble des noms de fichiers de la forme XuY, où u est un seul caractère
- Exemples

```
ll /*/*.d      tous les fichiers d'un rép de / qui se terminent par .d
ll -d /home/*  tous les sous-répertoires de /home
rm *          attention ! commande dangereuse, supprime tout le rép courant !
cp /lib/modules/*/*/*.* /home/toto  toto copie tous les pilotes dans son répertoire
personnel
cp /home/stage? /root/tmp
```

- Le modèle **[]** permet de sélectionner un élément de la liste ou de l'intervalle spécifié.
Le séparateur en ligne de commande étant l'espace, aucun espace ne doit être mis au début ou à la fin entre **[]**
- Plus précisément, un modèle de la forme **X [abc...z] Y** où X et Y peuvent être vides, désigne l'ensemble des noms de fichiers suivants: XaY, XbY ... XzY.
- Dans le cas d'une suite ordonnée de caractères comme abc ... z, on peut utiliser la notation intervalle **a-z**.
- On peut mélanger les deux notations, comme dans **[a-z].[0-9]**, ensemble des fichiers a.0, a.1, ..., b.0 b.1 etc ...
- Quelques exemples :
 - **ll a***
 - **ll [a-zA-Z]*** liste les fichiers du rép. courant dont le nom commence par a, b, c ou d minuscule ou majuscule (y compris les sous-rép.)
 - **cp ventes1[00-50].xls /home/toto/bilan** copie tous les fichiers ventes100.xls jusqu'à ventes150.xls
 - **lpr ~toto/formation/plan9[345].html** imprime les 3 fichiers plan93.html, plan94.html, ..



TP 3

Etudier et commenter les commandes suivantes, en étant connecté **root**

Commande Signification de cette commande ? que remarquez vous ?

```
ll ~/m*
cd
ll *.*          où sont passés les autres fichiers ?
ll *           que viennent ici faire les répertoires ?
ll *i*
ll [a-n]*
ll [an]*       quelle différence ?
ll [^an]* | less
ll *.*htm*
ll [a-z]*/*.pl
mkdir ~
```

lister tous les répertoires dont le nom commence par stage, avec une variable

```
user=stage
echo $user
ll -d home/$user*
```

Les commandes du shell

référence : **man bash**

Analyse de la ligne de commande

- Le shell commence par découper la ligne en mots séparés par des blancs.
Le premier mot attendu est le nom d'une commande. Les mots suivants sont considérés comme des paramètres dont la "compréhension" incombe à la commande (ces paramètres ont-ils pour la commande la signification d'options, de noms de fichiers, etc ...). **Donc la syntaxe à appliquer aux paramètres dépend de la commande.**

- Voici un exemple : supposons les comptes `stagex`, `x=1..9` déjà créés.
`grep -n stage. /etc/passwd` La commande **grep** attend des options précédées de -, puis un modèle (expression rationnelle) des chaînes à chercher, et enfin un ensemble de fichiers où elle doit chercher.



TP 4

- `grep -n sta /etc/passwd` ---> recherche dans le fichier `/etc/passwd` la sous-chaîne **sta**, en indiquant les N° de lignes (option `-n`)
- `grep -nw sta /etc/passwd` ---> recherche ... (l'option `-w` impose la recherche d'un mot entier, et pas d'une sous-chaîne)
- `grep -nw stage. /etc/passwd` ---> recherche ...
- `grep -nw stage? /etc/passwd` ---> quelle signification pour `grep` du ?
- `grep -nw stage? /etc/*` --->
- `grep -n ftp* /etc/rc.d/init.d/*` -->

Valeur de retour d'une commande

- Chaque commande transmet au programme appelant un code, appelée valeur de retour (*exit status*) qui stipule la manière dont son exécution s'est déroulée.
- Par convention du shell BASH, **la valeur de retour est toujours 0 si la commande s'est déroulée correctement**, sans erreur (attention, c'est l'inverse du langage C !)
- Une valeur de retour différente de 0 signale donc une erreur, qui peut être éventuellement analysée selon cette valeur.
- Une variable système spéciale **\$?** contient toujours la valeur de retour de la précédente commande. On peut afficher cette valeur avec la commande **echo**
Exemples :

```
[toto@p00]$ ll ~
[toto@p00]$ echo $?          --> 0
[toto@p00]$ ll /root
[toto@p00]$ echo $?          --> 1, si toto n'est pas root !
```

Enchaînement des commandes

- Habituellement, une ligne de commande saisie au prompt de la console ou bien écrite dans un script est une phrase composée de mots séparés par des espaces (ou des tabulations); le premier mot est considéré comme le nom d'une commande et le shell cherche à l'exécuter; les mots suivants sont des options ou paramètres de cette commande.
- Pour inhiber cette interprétation des espaces, il faut entourer le groupe de mots de quotes ou de guillemets, ce groupe sera alors interprété comme un seul paramètre.
Exemple : recherche de la chaîne *jules toto* (qui constitue un seul paramètre) sur les lignes de `/etc/passwd` (l'option `-i` pour s'affranchir de la casse)
`grep -i "jules toto" /etc/passwd`
- En général, on place une commande par ligne que ce soit en ligne de commande ou dans un script.
Le point-virgule `;` a le rôle de séparateur de séquence **inconditionnel**.
Il permet ainsi d'écrire une séquence de plusieurs commandes sur une même ligne.
Toutes les commandes sont inconditionnellement exécutées (même si l'une d'entre elle provoque une erreur), et leur résultats respectifs sont envoyés sur la sortie standard, séparés par un retour à la ligne `"\n"`.
On peut connaître la valeur de retour de chacune en interrogeant la variable **\$?**



TP 5

Si `toto6` n'est pas un utilisateur valide ?

```
[root@p00]$ grep toto6 /etc/passwd ; echo $?
```

*le groupe root existe déjà, il ne peut pas être recréé,
prévoir les codes de retour*

```
[root@p00] # $ who am i; echo $?; groupadd root; echo $?; date; echo $?
```

Enchaînement conditionnels des commandes

- Les séparateurs **&&** et **||** sur la ligne de commande sont des séparateurs qui jouent les rôles d'opérateurs **conditionnels**, en ce sens que la 2ème commande sera exécutée en fonction du code de retour de la 1ère commande.
- Dans `commande1 && commande2`, `commande2` ne sera exécutée que si le code de retour de `commande1` est 0 (exécution correcte)
Dans `commande1 || commande2`, `commande2` ne sera exécutée que si le code de retour de `commande1` est différent de 0 (exécution erronée)
- Exemples : trouver leur signification

```
cd ~/tmp || mkdir $HOME/tmp  
extrait de /etc/rc.d/inet.d/inetd  
[ -f /usr/sbin/inetd ] || exit 0
```

Redirections des entrées-sorties

Toutes les commandes (du noyau, du shell et créées par le programmeur) sont dotées par le système de **3 canaux de communication** :

- entrée standard (`stdin`=standard input) pour lire des données,
- la sortie standard (`stdout`) pour envoyer des résultats
- et la sortie des erreurs (`stderr`).
- Par défaut les canaux d'entrées et de sorties communiquent avec le clavier et l'écran : les commandes et les programmes qui ont besoin de données les attendent en provenance du clavier et expédient leurs résultats pour affichage sur le moniteur.
- Il est possible de les détourner pour les rediriger vers des fichiers ou même vers les entrées-sorties d'autres commandes.
Les symboles utilisées sont :

- `<` redirection de l'entrée standard à partir d'un fichier (et non depuis le clavier)
- `>` redirection de la sortie standard en direction d'un fichier (et non vers l'écran clavier)
attention ! le fichier est créé .. et écrase sans préavis le fichier existant portant le même nom.
- `>>` redirection de la sortie standard **à la fin** du fichier s'il existe déjà
- `|` (= alt 124) enchaînement de commandes (appelé aussi **tube** ou **pipe**)
la sortie de la commande gauche est envoyée en entrée de la commande droite
Fréquemment utilisé avec **less** (ou `more`) pour examiner l'affichage sur le moniteur.
La valeur de retour est celle de la dernière commande.

- Tester
ll --help | less

TP 6

Etudier et commenter les commandes suivantes

- Lorsqu'une commande attend une entrée clavier , taper quelques lignes (du texte qcq) puis terminer par `Ctrl-d` (symbole EOF=end-of-file) pour sauvegarder.
(une liste de symboles ligne de commandes est obtenue par `stty -a`)
- On repasse en mode commande ... et on envoie la commande suivante.
- `lpr` est la commande d'impression sur la file d'attente par défaut.

- `wc` (=word count) compte le nombre de lignes, de mots et de caractères du fichier en entrée (suivant les options `-l`, `-w`, `-c`).

1. Exemples :

```
cd
cat > essai.txt
cat essai.txt
sort < essai.txt
cat >> essai.txt
sort < essai.txt
sort < essai.txt > essai-tri.txt
cat essai-tri.txt
cat essai.txt essai-tri.txt
```

2. Quel est l'effet de la commande suivante ? Vérifiez (essai.txt est le fichier créé précédemment)

```
wc -w < essai.txt > mots.txt
Que se passe t-il si on enlève l'option -w ?
```

3. Pour obtenir le même affichage final, remplacez la séquence suivante par une seule commande

```
cd /etc
ll > /tmp/liste.txt
cat /tmp/liste.txt
wc -l < /tmp/liste.txt
```

4. `ll`

```
ll /etc | less
ll | sort
ll | wc -l
```

5. `who`

```
who | sort
cat | sort > essai-pipe.txt
Pouvez-vous prévoir la différence entre :
cat essai.txt | lpr
cat essai.txt > lpr
```

6. Enregistrer dans un même fichier **truc.txt**, la liste des utilisateurs actuellement connectés, la date du jour, le nom de l'utilisateur actif et le rép. personnel trié.

Substitution de commande

- Ce procédé permet de substituer au texte d'une commande le résultat de son exécution qui est envoyé sur la sortie standard. La commande simple ou complexe (avec redirections, tubes) doit être entourée de l'opérateur antiquote **`** **Alt-Gr7** ou être placée dans un parenthésage précédé de **\$(...)**. D'une manière générale, il est recommandé d'entourer l'expression de " "
- Exemple :

```
echo "`whoami`, nous sommes le `date` "
attention, pas d'espace entre $ et (
echo "$(whoami), nous sommes le $(date) "
```



TP 7

- Comparer :

```
pwd
echo pwd
echo `pwd`
echo "Il y a `ls | wc -l` fichiers dans `pwd` "
```

- `tr 'A-Z' 'a-z'` traduit chaque caractère majuscule reçu sur son entrée en l'équivalent minuscule . Que réalise alors cette commande ?
`echo -n "Votre mot de passe ?"`
`read mdp`
`mdp = $(echo $mdp | tr 'A-Z' 'a-z')`
- Les substitutions de commande peuvent être imbriquées.
Attention à bien placer les " ". Exemple :
`echo "Nombre de fichiers du répertoire personnel : $( ls -l $( pwd ))" | less`
- Si on connaît `grep` et `cut`, quelle est la signification de :

```
nom=toto  
numero=$(cat /etc/passwd | grep -wi "^$nom" | cut -d: -f3)
```

Extraction de texte

Voir [filtres tr, grep, cut, sed ..](#)



Commandes de filtres : grep, sed, awk

Généralités

- Un filtre est une commande qui lit les données sur l'entrée standard, effectue des traitements sur les lignes reçues et écrit le résultat sur la sortie standard.
- Bien sûr les entrées/sorties peuvent être redirigées, et enchainées avec des tubes.
A noter que le caractère d'indirection < en entrée n'est pas obligatoire pour les filtres
Ainsi, dans `# cat /etc/*.conf > tous.conf` cat va bien lire les fichiers qui correspondent au modèle `/etc/*.conf` et les concaténer dans le fichier `tous.conf`
- Dans ce chapitre, on va revoir ou découvrir les principaux filtres utilisés dans le monde UNIX
 - [cat, more et less](#)
 - [grep](#)
 - [cut](#)
 - [wc](#)
 - [tr](#)
 - [sed](#)
 - [awk](#)

Les commandes cat, more et less

`cat f1 f2 .. > f` concatène f1, f2 .. dans le nouveau fichier f
`less f1 f2 .. > f` concatène les fichiers f1 f2 .. en un seul fichier f (comme cat)
`less f3 >> f` ajoute le contenu du fichier f3 à la suite du contenu du fichier f

La commande grep : sélection de lignes

Cet utilitaire (*General Regular Expression Parser*, analyseur général d'expression régulière) sélectionne toutes les lignes qui satisfont une expression régulière (ou rationnelle).

Syntaxe

grep [options] expreg [fichiers]

Cette commande recherche dans les fichiers ou sur son entrée standard des lignes de texte qui satisfont l'expression régulière expreg indiquée.

Sa sortie peut être redirigée dans un fichier.

options

- c donne seulement le nombre de lignes trouvées obéissant au critère
- l donne seulement le nom des fichiers où le critère a été trouvé
- v donne les lignes où le critère **n'a pas** été trouvé
- i ne pas tenir compte de la casse (ne pas différencier majuscules minuscules)
- n pour n'afficher que les numéros des lignes trouvées
- w pour imposer que le motif corresponde à un mot entier d'une ligne

constructions

grep est souvent inclus dans un tube qui lui fournit en entrée le fichier à étudier.

Exemple : quelle est la question posée ?

```
cat /etc/passwd | cut -d: -f1 | grep -w "jean" > sortie
```

Expressions reconnues

Grep ne reconnaît pas toutes les [expressions rationnelles](#) étendues.

Voici la liste des symboles utilisables par **grep** : `.` `*` `[]` `^` `$`

- `.` signifie un caractère quelconque
- `*` répétition du caractère situé devant
- `^` début de ligne
- `$` fin d'une ligne (donc `"e$"` mots se terminant par e)
- `[...]` contient une liste ou un intervalle de caractères recherchés
- `[^...]` caractères interdits.

Attention

Pour éviter une confusion entre les interprétations de ces symboles spéciaux par **grep** ou par le shell, il est indispensable de "verrouiller" **expreg** en plaçant l'expression entre guillemets `" "` (et non entre quotes `!`).



Exemples

Etudier et commenter les commandes suivantes :

1. *cherche dans fichier, les lignes dont la 1ère lettre est qcq et la 2ème doit être o*
`grep "^o" fichier`
2. *cherche dans le fichier passwd les lignes commençant par t*
`grep "^t" /etc/passwd`
3. *cherche les lignes ne commençant pas commençant par t*
`grep -w "^t" /etc/passwd`
4. *cherche les lignes contenant les mots suivant le modèle T.t.*
`grep "T.t." /etc/passwd`
5. *cherche dans le fichier des groupes, ceux qui commencent par a ou b .. ou j*
`less /etc/group | grep "[a-j]"`
6. *pour lister les s-répertoires du rép. /etc*
`ll /etc | grep "^d"`
7. *compter les lignes saisies au clavier qui se termine par a*
`grep -c "a$"`
8. *afficher les lignes des fichiers essai?.txt qui contiennent a, b ou c*
`grep [abc] "essai?.txt"`
9. *détourne le flot de sortie du moniteur pour l'envoyer sur l'entrée de wc pour ?*
`grep [abc] "essai?.txt" | wc -l`

Exercices

1. donner une version sans `cut` de la commande précédente
2. *Comment ne sélectionner que "root" avec*
`cat /etc/passwd | cut -d : -f 1 | grep "r"`
3. on sait que `ps aux` donne la liste des processus. La commande `/etc/X11/X` est celle qui lance le serveur X. Il est nécessaire de connaître son **PID**, en cas de plantage du serveur X
Ecrire la commande qui retourne la ligne correspondante.
4. Se placer dans `/etc`. En une seule commande, faire calculer et afficher le nombre de sous-répertoires de `/etc`, sous la forme :
"Il y a 33 répertoires dans /etc"

5. Créer un fichier `essai1` contenant quelques lignes dont des lignes vides Avec **grep** générer le fichier `essai2` à partir de `essai1` sans ligne vide

```
cat essai1 ..... >essai2
```

Réponses

cut : sélection de colonnes

La commande **cut** présente 2 formes suivant que l'on sélectionne des colonnes de caractères ou qu'on distingue des champs séparés par un caractère précis.

sélection colonne

cut -c(sélection_colonnes) [fichiers]

Exemples

- *affiche le 5ième caractère*
`cut -c5 fichier`
- *affiche du 5ième **au** 10ème caractères*
`cut -c5-10 fichier`
- *affiche le 5ième **et** le 10ème caractères*
`cut -c5-10 fichier`
- *affiche à partir du 5ième (jusqu'à la fin)*
`cut -c5- fichier`

sélection champs

cut -d(séparateur) -f(sélection_champs) [fichiers]



Exercices

Etudier les commandes suivantes

1. `cut -d":" -f1,6 /etc/group`
2. *Que réalise la ligne suivante ? Vérifiez*
`grep "^st" /etc/passwd | cut -d":" -f1,3-4,6`

La commande wc

Exemples

```
cat /etc/passwd | grep /bin/bash/ | wc -l
pour compter les titulaires d'un compte pouvant se connecter avec le login shell
less /etc/passwd | grep -vc /bin/bash/
négation de la question précédente (revoir les rôles ds options -c et -v)
```

La commande tr

tr=Translate, est un filtre ne reconnaissant pas les expr. régulières.

Cette commande est le plus souvent associée à des redirections

Les caractères entrés sont traités et le résultat est envoyé sur la sortie standard

On peut utiliser les intervalles du type `a - z` et les codes ASCII des caractères en notation octale `\0xx`

Syntaxe

1. **tr [options] ch1 ch2 <fich1 >fich2**

Remplace toutes les occurrences de TOUS les caractères de ch1 par le caractère de ch2, de même rang, dans le flot d'entrée.

2. Exemple

```
# pour convertir et afficher la ligne saisie au clavier en minuscules
read ligne; echo $ligne | tr 'A-Z' 'a-z'
```

3. **tr -c chaîne car** remplace tout caractère NON INCLUS dans la chaîne chaîne par le caractère car

```
# remplace supprime tous les caractères différents de a,b, ..z par un espace
echo $ligne | tr -c a-z ' '
```

4. **tr -d chaîne** supprime tout caractère entré, appartenant à la chaîne chaîne

```
# supprime toutes les minuscules non accentuées
echo $ligne | tr -d a-z
```

5. **tr -s chaîne** supprime toute répétition des caractères contenus dans chaîne

```
# supprime les espaces multiples entre les mots
echo $ligne | tr -s ' '
```



Exercices

Expliquer les effets de ces commandes :

1. `tr 'a,/' 'A;_' <fich1 >fich2`
2. `tr 'a-z' 'A-Z' <fich1 >fich2`
3. `tr -d '\011\015\032' <fich1 >fich2`
4. `tr -s '\011\012' <fich1 >fich2`
5. `tr -cs 'a-zA-Z0-9' '\n' <fich1 | sort | uniq >fich2`

Réponses

L'utilitaire sed

Il s'agit d'un utilitaire (*sed* = "*Stream EDitor*") qui sélectionne les lignes d'un fichier texte (ou d'un flot provenant d'un pipe) vérifiant une expression régulière et qui leur applique un traitement ou un remplacement.

Syntaxe

`sed [-n] [-e script] [-f fichier-commandes] fichier-source`

- L'option -n empêche la sortie à l'écran du résultat (souvent associé à l'option p)
- Le fichier source est traité ligne par ligne conformément à la liste des commandes (-e) ou au fichier de commandes (-f)

Commande de substitution

- La commande s permet d'effectuer des substitutions suivant la syntaxe :
sed [adresse]s/expr-régulière/remplacement/options
- Attention ! contrairement à ce que l'on pourrait attendre, cette commande laisse passer toutes les lignes et ne sélectionne pas celles qui ont satisfait l'expression régulière et donc subi la substitution. Pour sélectionner, voir la commande de destruction.
- Options

- Sans précision, la commande ne s'applique qu'à la 1ère occurrence de chaque ligne.
- 0...9 : indique que la substitution ne s'applique qu'à la nième occurrence
- g : effectue les modifications sur toutes les occurrences trouvées.
- Exemple : `sed s/moi/toi/g fich.moi > fich.toi`
le fichier `fich.moi` est parcouru, à chaque occurrence de "moi", ce mot est remplacé par "toi" et le nouveau fichier est sauvegardé sous le nom `fich.toi`
- Délimiteur
Le slash / étant très utilisé au niveau du shell comme séparateur de niveau de répertoire, il est possible d'utiliser à la place tout autre caractère comme #
`sed s#/home#/rep_perso#g /etc/passwd > /tmp/passwd.new`

Destruction ou sélection

- Cette option permet de filtrer les lignes qui satisfont une expression régulière. Ces lignes ne sont pas détruites dans le fichier d'origine, mais ne sont pas transmises en sortie.
- Comment modifier alors le fichier à traiter ?
`cp fichier copie`
`sed /.../d copie`
- Par exemple, pour détruire toutes les lignes vides d'un fichier :
`sed /^$/d`

Ajout, insertion et modification

Pour utiliser ces commandes, il est nécessaire de les saisir sur plusieurs lignes

```
sed [adresse] commande\
expression
```

La commande peut être :

```
a pour ajout ;
i pour insertion ;
c pour modification.
```

L'utilitaire awk

Son nom vient de ses 3 auteurs Aho, Weinberger et Kernighan C'est l'implémentation GNU du langage awk, petit langage interprété par le programme awk.

A suivre ...

Réponses

grep

1. `less /etc/group | grep "^[a-j]"`
2. `ps aux | grep "/etc/X11/X"`
3. `catessai1 | grep -v "^$" >essai2`

tr

1. remplace resp. a , et / par A ; et _

2. met en majuscules
3. suppression des caractères ASCII tab (9), retour-chariot (13) et Ctrl-Z (26)
4. suppression des répétitions des espaces, des tab et des passages à la ligne (10)
5. découpage en mots de la chaîne entrée, trie des lignes et suppression des doublons



Introduction aux scripts BASH

Instructions [test](#) [if](#) [case](#) [for](#) [while](#) [select](#)

La programmation shell

- Un script bash est fichier de type texte contenant une suite de commandes shell, exécutable par l'interpréteur (ici le programme `/bin/bash`), comme une commande unique. Un script peut être lancé en ligne de commande, comme dans un autre script.
- Mais il s'agit bien plus qu'un simple enchainement de commande : on peut définir des variables et utiliser des structures de contrôle, ce qui lui confère le statut de langage de programmation interprété et complet.
- Le langage **bash** gère notamment :
 - la gestion des entrées-sorties et de leur redirection
 - des variables définies par le programmeur et des variables systèmes
 - le passage de paramètres
 - des structures conditionnelles et itératives
 - des fonctions internes

Saisie du script

- Utiliser `vi` de préférence à `mc` qui ne traite pas les accents (mais `mc` est bien pratique !)
- Les lignes commençant par le caractère dièse `#` sont des commentaires. En insérer abondamment !
- Le script doit débuter par l'indication de son interpréteur écrite sur la première ligne : **`#!/bin/bash`**.
En fait si le shell par défaut est `bash`, cette ligne est superflue
- Exemple

```
#!/bin/bash
# script bonjour
# affiche un salut à l'utilisateur qui l'a lancé
# la variable d'environnement $USER contient le nom de login
echo ---- Bonjour $USER ----
# l'option -n empêche le passage à la ligne
# le ; sert de séparateur des commandes sur la ligne
echo -n "Nous sommes le " ; date
# recherche de $USER en début de ligne dans le fichier passwd
# puis extraction de l'uid au 3ème champ, et affichage
echo "Ton numéro d'utilisateur est " $(grep "^$USER" /etc/passwd | cut -d: -f3)
```

Exécution du script

- Il est indispensable que le fichier script ait la permission **x** (soit exécutable). Lui accorder cette permission pour tous ses utilisateurs avec `chmod` :
`chmod a+x bonjour`
- Pour lancer l'exécution du script, taper **`./bonjour`**, `./` indiquant le chemin, ici le répertoire courant. Ou bien indiquer le chemin absolu à partir de la racine. Ceci dans le cas où le répertoire contenant le script n'est pas listé dans le `PATH`
- Si les scripts personnels sont systématiquement stockés dans un rép. précis, par exemple `/home//bin`, on peut ajouter ce chemin dans le `PATH`.
Pour cela, il suffit d'ajouter la ligne suivante dans `/etc/skel/.bash_profile`, qui est recopié dans chaque répertoire dont le rôle est d'affiner le profil personnel du shell de chaque utilisateur.
`# bash_profile`

```
.....
#user specific environment and statup programs
PATH=$PATH:$HOME/bin
```

- Mais on peut plus simplement s'initier au langage Bash, directement en dialoguant avec l'interpréteur.
Si on entre une instruction incomplète en ligne de commande, l'interpréteur passe à la ligne suivante en affichant le prompt **>** et attend la suite de l'instruction (pour quitter Ctrl-C).

- Mise au point, débogage

Exécution en mode "trace" (-x) et en mode "verbeux" (-v) **sh -x ./bonjour** Pour aider à la mise au point d'un script, on peut insérer des lignes temporaires : **echo \$var** pour afficher la valeur de la variable

exit 1 pour forcer l'arrêt du script à cet endroit

- On peut passer des arguments à la suite du nom du script, séparés par des espaces. Les valeurs de ces paramètres sont récupérables dans le script grâce aux paramètres de position \$1, \$2 .. mais, contrairement aux langages de programmation classiques, ils ne peuvent pas être modifiés.

Exemple

```
#!/bin/bash
# appel du script : ./bonjour nom prenom
if [ $# = 2 ]
then
echo "Bonjour $2 $1 et bonne journée !"
else
echo "Syntaxe : $0 nom prenom"
fi
```

Entrées-sorties

Ce sont les voies de communication entre le programme bash et la console :

- **echo**, affiche son argument texte entre guillemets sur la sortie standard, c-à-d l'écran.
La validation d'une commande **echo** provoque un saut de ligne.
echo "Bonjour à tous !"
- On peut insérer les caractères spéciaux habituels, qui seront interprétés seulement si l'option **-e** suit **echo**
\n (saut ligne), **\b** retour arrière), **\t** (tabulation), **\a** (alarme), **\c** (fin sans saut de ligne)

```
echo "Bonjour \nà tous !"
echo -e "Bonjour \nà tous !"
echo -e "Bonjour \nà toutes \net à tous ! \c"
```

- **read**, permet l'affectation directe par lecture de la valeur, saisie sur l'entrée standard au clavier
read var1 var2 ... attend la saisie au clavier d'une liste de valeurs pour les affecter, après la validation globale, respectivement aux variables **var1**, **var2** ..

```
echo "Donnez votre prénom et votre nom"
read prenom nom
echo "Bonjour $prenom $nom"
```

Les variables BASH

Variables programmeur

De façon générale, elles sont de type texte. On distingue les variables définies par le programmeur et les variables systèmes

- syntaxe : **variable=valeur**
Attention ! le signe = NE DOIT PAS être entouré d'espace(s)
On peut initialiser une variable à une chaîne vide :

chaîne_vide=

- Si **valeur** est une chaîne avec des espaces ou des caractères spéciaux, l'entourer de " " ou de ''
- Le caractère \ permet de masquer le sens d'un caractère spécial comme " ou '.

```
chaîne=Bonjour à tous
echo $chaîne
```

- Référence à la valeur d'une variable : faire précéder son nom du symbole \$
- Pour afficher toutes les variables : **set**
- Pour empêcher la modification d'une variable, invoquer la commande **readonly**
- Substitution de variable

Si une chaîne contient la référence à une variable, le shell doit d'abord remplacer cette référence par sa valeur avant d'interpréter la phrase globalement. Cela est effectué par l'utilisation de " ", dans ce cas obligatoire à la place de ' '.

Exemples

```
n=123 ;
echo "la variable \n vaut $n"
salut="bonjour à tous !"
echo "Alors moi je dis : $salut"
echo 'Alors moi je dis : $salut'
echo "Alors moi je dis : \"$salut\" "
readonly salut
salut="bonjour à tous, sauf à toto"
echo "Alors moi je dis : $salut"
```

- Variables exportées

Toute variable est définie dans un shell. Pour qu'elle devienne globale elle doit être exportée par la commande :

```
export variable
```

export --> Pour obtenir la liste des variables exportées

- Opérateur {} dans les variables

Dans certains cas en programmation, on peut être amené à utiliser des noms de variables dans d'autres variables. Comme il n'y a pas de substitution automatique, la présence de {} force l'interprétation des variables incluses.

Voici un exemple :

```
user="/home/stage"
echo $user
u1=$user1
echo $u1    --> ce n'est pas le résultat escompté !
u1=${user}1
echo $u1
```

Variables d'environnement

Ce sont les variables systèmes dont la liste est consultable par la commande **env** | **less**

Les plus utiles sont \$HOME, \$PATH, \$USER, \$PS1, \$SHELL, \$ENV, \$PWD ..



Exemple (bien sûr on est pas forcément connecté sous le pseudo toto)

```
[toto@pxx toto]$ moi=Toto
[toto@pxx toto]$ p="Je m'appelle $moi"
[toto@pxx toto]$ echo Aujourd'hui, quel jour sommes nous ? ; read jour
[toto@pxx toto]$ echo aujourd'hui $jour, $p sous le nom $USER,
est connecté à la station $HOSTNAME
```

Variables prédéfinies spéciales

Elles sont gérées par le système et s'avèrent très utiles dans les scripts. Bien entendu, elles ne sont accessibles qu'en lecture.

Ces variables sont automatiquement affectées lors d'un appel de script suivi d'une liste de paramètres. Leurs valeurs sont récupérables dans \$1, \$2 ... \$9

\$?	C'est la valeur de sortie de la dernière commande. Elle vaut 0 si la commande s'est déroulée sans pb.
\$0	Cette variable contient le nom du script
\$1 à \$9	Les (éventuels) premiers arguments passés à l'appel du script
\$#	Le nombre d'arguments passés au script
\$*	La liste des arguments à partir de \$1
\$\$	le n° PID du processus courant
\$!	le n° PID du processus fils

```
ls -l
echo $?      ----> 0
ifconfig ttyS1
echo $?      ---> 1
```

Passage de paramètres

On peut récupérer facilement les compléments de commande passés sous forme d'arguments sur la ligne de commande, à la suite du nom du script, et les utiliser pour effectuer des traitements.

Ce sont les variables système spéciales **\$1** , **\$2** **\$9** appelées paramètres de position.

Celles-ci prennent au moment de l'appel du script, les valeurs des chaînes passées à la suite du nom du script (le séparateur de mot est l'espace, donc utiliser si nécessaire des "").

A noter que :

- le nombre d'argument est connu avec \$#
- la liste complète des valeurs des paramètres (au delà des 9 premières) s'obtient avec \$*
- le nom du script rest recopié dans \$0

La commande shift

- Il n'y a que 9 paramètres de position de \$1 à \$9, et s'il y a davantage de paramètres transmis, comment les récupérer ?
- **shift** effectue un décalage de pas +1 dans les variables \$: \$1 prend la valeur de \$2, etc...
- Exemple


```
a=1 ; b=2 ; c=3 ; set a b c
echo somme10 1 2 3 4 5 6 7 8 9 10
echo $1, $2, $3
```

La commande set

-
-
- Exemple


```
a=1 ; b=2 ; c=3
set a b c
echo $1, $2, $3
# les valeurs de a, b, c sont récupérées dans $1, $2, $3
```

La commande test

Généralités

Comme son nom l'indique, elle sert à vérifier des conditions. Ces conditions portent sur des fichiers (le plus souvent), ou des chaînes ou une expression numérique.

Cette commande courante sert donc à prendre des (bonnes) décisions, d'où son utilisation comme condition dans les structures conditionnelles `if .. then .. else`, en quelque sorte à la place de variables booléennes ... qui n'existent pas.

Syntaxe

- `test expression`
- `[ expression ]` attention aux espaces autour de `expression`

Valeur de retour

- **Rappels**
On sait que toute commande retourne une valeur finale au shell : 0 pour lui indiquer si elle s'est déroulée normalement ou un autre nombre si une erreur s'est produite.
Cette valeur numérique est stockée dans la variable spéciale `$?`
- La commande `test`, de même, retourne 0 si la condition est considérée comme vraie, une valeur différente de 0 sinon pour signifier qu'elle est fausse.

Tester un fichier

- Elle admet 2 syntaxes (la seconde est la plus utilisée) :

test option fichier
[option fichier]

- Tableau des principales options

option	signification quant au fichier
-e	il existe
-f	c'est un fichier normal
-d	c'est un répertoire
-r -w -x	il est lisible modifiable exécutable
-s	il n'est pas vide

- Exemples

```
[ -s $1 ]
vrai (renvoie 0) si le fichier passé en argument n'est pas vide
[ $# = 0 ]          le nombre d'arguments
est 0
[ -w fichier ]     le fichier est-il modifiable ?
[toto@p00]$ [ -r "/etc/passwd" ] toto peut-il lire le fichier /etc/passwd ?
[toto@p00]$ echo $? --> 0 (vrai)
[toto@p00]$ [ -r "/etc/shadow" ] toto peut-il lire le fichier /etc/shadow ?
[toto@p00]$ echo $? --> 1 (faux)
[toto@p00]$ [ -r "/etc/shadow" ] || echo "lecture du fichier interdite"
```

Tester une chaîne

- [option chaine]

option	signification
-z -n	la chaine est vide / n'est pas vide
= !=	les chaines comparées sont identiques différentes

- Exemples

```
[ -n "toto" ] ; echo $?  affiche la valeur renvoyée 0
ch="Bonjour" ; [ "$ch" = "bonjour" ] ; echo $?  affiche 1
[ $USER != "root" ] && echo "l'utilisateur n'est pas le \"root\" !"

```

Tester un nombre

- [nb1 option nb2]

Il y a d'abord un transtypage automatique de la chaine de caractères en nombre

option	signification
-eq -ne	égal différent
-lt -gt	strict. inf strict. sup
-le -ge	inf ou égal sup ou égal

- Exemples

```
a=15 ; [ "$a" -lt 15 ] ; echo $?

```

Opérations dans une commande test

-

option	valeur
[expr1 -a expr2]	(and) 0 si les 2 expr sont vraies
[expr1 -o expr2]	(or) 0 si l'une des 2 expr est vraie
[! expr1]	négation

- Exemples

```
Quel résultat ? envisager 2 cas ...
f="/root" ; [ -d "$f" -a -x "$f" ] ; echo $?
note=9; [ $note -lt 8 -o $note -ge 10 ] && echo "tu n'est pas convoqué(e) à l'oral"

```

Structures conditionnelles

```
if suite-de-commandes
then
# séquence exécutée si suite-de-commandes rend une valeur 0
  bloc-instruction1
else
# séquence exécutée sinon
  bloc-instruction2

```

```
fi
```

Attention ! si then est placé sur la 1ère ligne, séparer avec un ;

```
if commande; then
```

```
.....
```



Exemples

1. toto possède t-il un compte ? On teste la présence d'une ligne commençant par toto dans /etc/passwd (>/dev/null pour détourner l'affichage de la ligne trouvée)

```
if grep "^toto" /etc/passwd > /dev/null
then
  echo "Toto a déjà un compte"
fi
```
2. Si toto a eu une bonne note, on le félicite

```
note=17
if [ $note -gt 16 ] ---> test vrai, valeur retournée : 0
then echo "Très bien !"
fi
```
3. Avant d'exécuter un script, tester son existence.
 Extrait de \$HOME/.bash_profile

```
if [ -f ~/.bashrc ]
then
  ~/.bashrc
fi
```

Conditionnelles imbriquées

Pour imbriquer plusieurs conditions, on utilise la construction :

```
if commande1
then
  bloc-instruction1
elif commande2
then
  bloc-instruction2
else
# si toutes les conditions précédentes sont fausses
  bloc-instruction3
fi
```

Exemples

1. toto a t-il fait son devoir lisiblement ?

```
fichier=/home/toto/devoir1.html
if [ -f $fichier -a -r $fichier ]
then
  echo "je vais vérifier ton devoir."
elif [ ! -e $fichier ]
then
  echo "ton devoir n'existe pas !"
else
  echo "je ne peux pas le lire !"
fi
```
2. Supposons que le script exige la présence d'au moins un paramètre, il faut tester la valeur de \$#, est-elle nulle ?

```
if [ $# = 0 ]
then
echo "Erreur, la commande exige au moins un argument .."
exit 1
elif [ $# = 1 ]
then
echo "Donner le second argument : "
read arg2
fi
```

Choix multiples

```
case valeur in
  expr1) commandes ;;
  expr2) commandes ;;
  ...
esac
```



Exemples

- Supposons que le script doive réagir différemment selon l'utilisateur courant; on va faire plusieurs cas selon la valeur de **\$USER**

```
case $USER in
  root) echo "Mes respects M le $USER" ;;
  jean | stage?) echo "Salut à $USER" ;;
  toto) echo "Fais pas le zigo$USER \!" ;;
esac
```

- Le script attend une réponse *oui/non* de l'utilisateur

```
read reponse
case $reponse in
  [yYo0]*) ..... ;;
  [nN]*) .....;;
esac
```

- read langue

```
case $langue in
  francais) echo Bonjour ;;
  anglais) echo Hello ;;
  espagnol) echo Buenos Dias ;;
esac
```

- case \$param in

```
0|1|2|3|4|5|6|7|8|9 ) echo $param est un chiffre ;;
[0-9]*) echo $param est un nombre ;;
[a-zA-Z]*) echo $param est un nom ;;
*) echo $param de type non prévu ;;
esac
```

- Un vrai exemple, extrait du script **smb** (/etc/rc.d/init.d/smb)

```
# smb attend un paramètre, récupéré dans la variable $1
case "$1" in
  start)
    echo -n "Starting SMB services: "
    daemon smbd -D
    echo
    echo -n "Starting NMB services: "
```

```

    daemon nmbd -D
    ...;;
stop)
    echo -n "Shutting SMB services: "
    killproc smbd
    ....
esac

```

Structures itératives

Boucle for

- Syntaxe

```

for variable [in liste]
do
    commandes (utilisant $variable)
done

```

- Fonctionnement

Ce n'est pas une boucle **for** contrôlée habituelle fonctionnant comme dans les langages de programmation classiques (utiliser pour cela une boucle while avec une variable numérique).

La *variable* parcourt un ensemble de fichiers données par une liste ou bien implicitement et le bloc *commandes* est exécuté pour chaque de ses valeurs.

Les mots-clés **do** et **done** apparaissent en début de ligne (ou après un ;)

- La liste peut être explicite :

```

for nom in jean toto stage1
do
    echo "$nom, à bientôt"
done

```

- La liste peut être calculée à partir d'[une expression modèle](#)

```

# recopier les fichiers perso. de toto dans /tmp/toto
for fich in /home/toto/*
do
    cp $fich tmp/toto
done

```

- Si aucune liste n'est précisée, les valeurs sont prises dans la variable système \$@, c'est-à-dire en parcourant la liste des paramètres positionnels courants.

```

# pour construire une liste de fichiers dans $@
cd /home/stagex ; set * ; echo $@
for nom in $@
do echo $nom
done

```



- Expliquer les exemples suivants

```

for nom in /home/stage[1-9]
do
    echo "$nom, tout va bien ?"
done

```

○

○ for i in /home/*/* ; do echo \$i; done

- for i in /dev/tty[1-7] ; do setleds -D +numecho \$i; done
- for x in /home/st* do echo \$x >> liste-rep-stage.txt done less liste-rep-stage.txt
- for x in \$(grep "^st" /etc/passwd | cut -d: -f6) do echo \$x; echo \$x >> \$HOME/tmp/liste-rep-stage.txt done less liste-rep-stage.txt

Boucle while

while liste-commandes do commandes done	La répétition se poursuit TANT QUE la dernière commande de la liste est vraie (c-à-dire renvoie un <u>code de retour nul</u>)	Voici 2 exemples à comparer echo -e "Entrez un nom de fichier" read fich while [-z "\$fich"] do echo -e "Saisie à recommencer" read fich done
until liste-commandes do commandes done	La répétition se poursuit JUSQU'A CE QUE la dernière commande de la liste devienne vraie	while echo -e" Entrez un nom de fichier" read fich [-z "\$fich"] do echo -e "Saisie à recommencer" done

Exemples à tester

Pour dire bonjour toutes les secondes (arrêt par CTRL-C)

```
while true ;
do
echo "Bonjour M. $USER"
sleep 1
done
```

Lecture des lignes d'un fichier pour traitement : noter que la redirection de l'entrée de la commande while .. do .. done est placée à la fin

```
fich=/etc/passwd
while read ligne
do
echo $ligne
.....
done < $fich
```

Sortie et reprise de boucle

break placé dans le corps d'une boucle, provoque une sortie définitive cette boucle.

continue permet de sauter les instructions du corps de la boucle (qui suivent continue) et de "continuer" à l'itération suivante. Pour les boucles for, while et until, continue provoque donc la réévaluation immédiate du test de la boucle.

Exemples importants

Boucle de lecture au clavier arrêtée par la saisie de stop

```
#!/bin/bash
```

```
# syntaxe : lecture.sh
texte=""
while true
do
  read ligne
  if [ $ligne = stop ]
  then break
  else texte="$texte \n$ligne"
  fi
done
echo -e $texte

Lecture des lignes d'un fichier
fich="/etc/passwd"
grep "^stage" $fich | while true
do
  read ligne
  if [ "$ligne" = "" ] ; then break ; fi
  echo $ligne
done
```

Fonctions

- 2 syntaxes

```
function nom-fct {
  bloc d'instructions
}
nom-fct() {
  bloc d'instructions
}
```

- Exemple

En connexion root, on doit relancer les "démons", si on a modifié un fichier de configuration.

Par exemple /etc/rc.d/init.d/smb contient la commande `daemon smbd -D`, pourtant à l'essai `daemon` est une commande inconnue !

Reportons nous au début du fichier, le script /etc/rc.d/init.d/functions y est appelé. Celui-ci contient la fonction : `daemon() { .....`

- passage d'arguments

Le mécanisme est le même que vis à vis d'un script

- variables locales

Dans le corps de la fonction, on peut définir et utiliser des variables déclarées locales, en les introduisant avec le mot-clé **local**

Commandes diverses

Calcul sur les entiers relatifs

Ne pas confondre la syntaxe **`$((expression arithmétique))`** avec la substitution de commande **`$(commande)`**

Les priorités sont gérées par un parenthésage habituel

```
echo $((30+2*10/4))
echo $(( (30+2) * (10-7) /4 ))
```

tr

- Cette commande de filtre permet d'effectuer des remplacements de caractères dans une chaîne. Pour une étude plus complète voir le chapitre [filtres](#)
- Par exemple pour transformer une chaîne en minuscules

```
chaîne="Bonjour, comment allez VOUS aujourd'hui ?"
echo $chaîne | tr 'A-Z' 'a-z'
```
- Pour permettre l'utilisation de la commande **set** (voir ci-dessous), il est nécessaire que le séparateur de champ sur une ligne soit l'espace, et non pas par exemple :
 Exemple : créer un fichier passwd.txt qui introduit un espace à la place de ":" dans une **copie** de /etc/passwd

```
cat passwd | tr ":" " " > passwd.txt
```

set

Cette commande interne est très pratique pour séparer une ligne en une liste de mots, chacun de ces mots étant affecté à une variable positionnelle. Le caractère de séparation est l'espace.

```
# soit une chaîne ch qui contient une liste de mots
c="prof eleve classe note"
# set va lire chaque mot de la liste et l'affecter aux paramètres de position
set $c ; echo $1 $2 $3 $4
shift ; echo $1 $2 $3 $4
```

Le langage bash est inadapté aux calculs numériques. Mais si vraiment on veut calculer (sur des entiers) ..

Exemple : calcul des premières factorielles (attention, il y a rapidement un dépassement de capacité)

```
declare -i k ; k=1 ; p=1
while [ $k -le 10 ]
do echo "$k! = " $(p=$p * $k) ; k= $k+1
done
```

Idée (saugrenue !) : écrire le script somme-entiers.sh pour calculer la somme 1+2+...+n, où la valeur de n est passée en argument

eval

- Cette commande ordonne l'interprétation par le shell de la chaîne passée en argument. On peut ainsi construire une chaîne que l'appel à **eval** permettra d'exécuter comme une commande !
- Exemple

```
message="Quelle est la date d'aujourd'hui ?"
set $message
echo $# ---> le nombre de mots est 6
echo $4 ---> affiche la chaîne "date"
eval $4 ---> interprète la chaîne "date" comme une commande, donc ...
```

- Il est souvent pratique de construire une chaîne dont la valeur sera égale au libellé d'un enchaînement de commandes (par ;). Pour faire exécuter ces commandes contenues dans la chaîne, on la passe comme argument de la commande **eval**
- exemple 1

```
liste="date;who;pwd" ( ' ' ou " " obligatoires sinon le ; est un séparateur de commandes)
eval $liste
---> exécute bien les 3 commandes
```
- exemple 2
*Soit la chaîne \$user qui contient des information sur un compte à créer. S'il utilise un autre séparateur que ";" on fait appel à **tr** d'abord*

```
user="login=toto ; mdp=moi ; nom='Monsieur Toto' ; groupe=profs"
eval $user
echo $login $mdp $nom $groupe
```

```
useradd -G $groupe $login  
echo $mdp | (passwd --stdin $login)
```

Application aux scripts cgi

La soumission d'un formulaire HTML à la passerelle CGI est un mécanisme qui aboutit à la récupération par le script (Bash, Perl, etc..) d'une chaîne qui contient la requête sous un format particulier.

Voici un exemple

- Le professeur Toto a rempli un formulaire sur le WEB pour recevoir un spécimen, soient 2 champs de texte nommés en HTML **nom** et **prenom**, et a coché la case nommée **prof**
Voici la chaîne supposée nommée *requete* qui a été transmise :
`nom=toto&prenom=jules&prof=on`
- Cette chaîne contient toute l'information relative au formulaire saisi par l'utilisateur. Il s'agit toujours de la traiter pour en récupérer les couples (*var*, *valeur*) où *var* sont les noms donnés aux composants de formulaire et *valeur* les chaînes saisies ou exprimant une sélection.
Ce traitement écrit en Perl est élégant et élémentaire. [Voir ce petit exemple](#).
- En Bash le découpage de *\$chaîne*, puis les affectations des variables doit être fait "à la main"
`requete="nom=toto&prenom=jules&prof=on"`
*# le filtre **tr** va remplacer dans la chaîne \$requete qu'il reçoit, tous les caractères & par ;*

`commande=$(echo $requete | tr '&' ';')
echo $commande ---> nom=toto;prenom=jules;prof=on
eval $commande ---> exécute le ligne de commande, donc effectue les affectations !
echo $prenom $nom
[$prof = "on"] && echo "$prenom $nom est professeur"`

Traitement d'un formulaire en PERL

Requête CGI en PERL

Perl est un langage de script puissant et efficace pour traiter les fichiers texte.

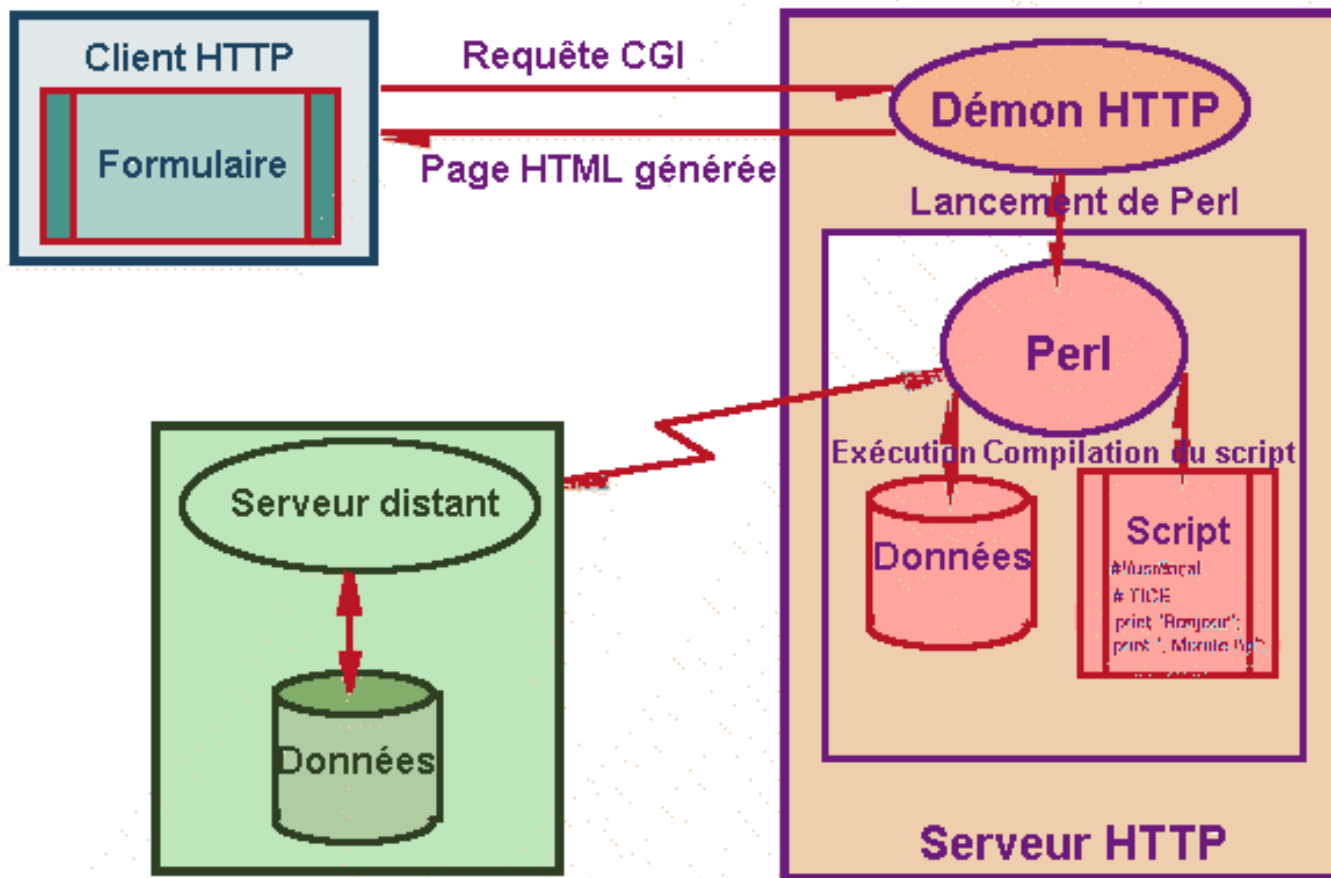
Il est très utilisé comme langage de script pour effectuer des [traitements CGI](#) sur le serveur.

Le code du script, proche du C, peut être adressé directement :

- dans un lien hypertexte URL du genre `<A HREF=http://www.serveur/cgi-bin/pgr.pl> ...`
- ou dans une soumission de formulaire `<FORM ACTION="http://www.serveur/cgi-bin/pgr.pl" METHOD=POST>`
- Il est compilé, puis exécuté par le moteur Perl.

Architecture générale

[image empruntée à ce remarquable site --> <http://www.ac-montpellier.fr/mafpen/tice/formation/perl.html>]



Requête CGI sur un script Perl

Les 2 méthodes de codage

Rappels

1. La méthode **GET** consiste à ajouter à l'URL la chaîne d'encodage des infos du formulaire. Cet URL est passé dans la variable **QUERY_STRING**
2. Pour la méthode **POST**, la même chaîne est expédiée sur l'entrée standard du script de CGI connecté à la

Scripts de décodage

méthode GET: script get.pl

```
#recupere le contenu de la variable d'environnement
$input = $ENV{"QUERY_STRING"};
#dissocie la chaine de caracteres en une liste
@liste= split(/&/,$input);
#parcours de la liste
foreach (@liste) {
    #dissocie la paire nom=valeur
    ($name,$value)= split(/=/, $_);
    #decode les valeurs
    $name =~ s/%(..)/pack("c",hex($1))/ge;
    $value =~ s/%(..)/pack("c",hex($1))/ge;

    #Traitement des données ....
}
```

méthode POST : script post.pl

```
#recupere le contenu du buffer de l'entrée standard STDIN
$in = <STDIN>;
#supprime les deux CRLF inseres par le protocole HTTP
chop($in);
chop($in);
#dissocie la chaine de caractere en une liste
@liste = split(/&/,$in);
#parcours de la liste
foreach(@liste) {
    #dissocie la paire nom=valeur
    ($nom,$valeur)= split(/=/, $_);
    #decode les valeurs
    $nom =~ s/%(..)/pack("c",hex($1))/ge;
    $valeur =~ s/%(..)/pack("c",hex($1))/ge;

    #Traitement des donnees ...
}
```

Exemple de traitement d'un formulaire en PERL

Le formulaire suivant est situé dans le WEB du serveur p00

- inclus dans le fichier
/home/httpd/html/form/formu.html
- accessible à l'URL : <http://p00/formu.html>

Formulaire

Indiquer :

Nom

Prénom

Sexe féminin masculin

Profession enseignant formateur

Code du formulaire

```
<FORM ACTION="http://p00/cgi-bin/formu2.pl" METHOD=POST>
<H1><DIV ALIGN="center">Traitement de formulaire par PERL</DIV></H1>
Indiquez :
Nom <INPUT TYPE="text" NAME="nom" SIZE=25 value="Toto">
Prénom <INPUT TYPE="text" NAME="prenom" SIZE=25 value="Jules">
Sexe    <INPUT TYPE="radio" NAME="sexe" VALUE="féminin"> féminin
         <INPUT TYPE="radio" NAME="sexe" VALUE="masculin" CHECKED> masculin
Profession <input type="radio" checked NAME="profession" VALUE=enseignant>
enseignant
         <input type="radio" NAME="profession" VALUE="formateur" > formateur
<P>
<INPUT TYPE="submit" VALUE="Valider">
<INPUT TYPE="reset" VALUE="Effacer">
</FORM>
```

Le script Perl

La validation du formulaire par clic sur le bouton submit provoque l'appel au script **formu2.pl**, écrit en PERL et situé réellement sur le serveur **p00** à **/home/httpd/cgi-bin/formu.pl**

```
#!/usr/bin/perl
```

```
# exécution de /home/httpd/cgi-bin/formu.pl
```

```
# récupère l'entrée standard dans la variable $in
read(STDIN, $in, $ENV{CONTENT_LENGTH});
```

```
# la chaîne $in est coupée suivant le caractère & et crée la liste @champs
@champs = split(/&/,$in);
```

```
# traitement de chaque élément $e de la liste @champs
foreach $e (@champs) {
```

```
  # dissocie chaque élément, de la forme nom=valeur,
  # en une paire de variable (nom,valeur)
  ($nom, $valeur) = split(/=/,$e);
```

```
  # transforme tous les caractères saisis en minuscules
  $valeur =~ tr/A-Z/a-z/;
```

```

# crée à partir du tableau @champs,
# une liste associative %champs
$champs{$nom}=$valeur;
}
# génère l'en-tête du document HTML renvoyé
print("Content-Type: text/html\n\n");
# puis le document HTML
print <<"SORTIE";
<HEAD><TITLE> Réponse </TITLE></HEAD>
<BODY>
<H2 ALIGN=CENTER>Réponse au questionnaire</H2>
<CENTER><TABLE BORDER><TR> <TH>Nom du champ <TH>Valeur</TR>
SORTIE

# le traitement est ici réduit à afficher les valeurs transmises
while (($nom, $valeur) = each(%champs)) {
print "<TR><Td>$nom = </td><Td>$valeur</td></TR>";
}
print "</TABLE></CENTER></BODY>";

```

Erreurs rencontrées

Message d'erreur renvoyé :

Internal Server Error

The server encountered an internal error or misconfiguration and was unable to complete your request.

Please contact the server administrator, root@localhost and inform them of the time the error occurred, and anything you might

have done that may have caused the error.

Premature end of script headers: /home/httpd/cgi-bin/post.pl

Attention !

- La 1ère ligne #!/usr/bin/perl ne doit pas être précédée d'espace
- Il faut éviter d'écrire les scripts sur plate-forme DOS, (puis les transférer sur Linux) car les caractères retour-chariot (dos) et \n (linux) ne correspondent pas. L'édition avec **vi** révèle des **^M** à supprimer.



TP1 Scripts BASH

I. Scripts a: et c:

Pour les nostalgiques du DOS, il s'agit de taper **a** : en ligne de commande et d'obtenir le contenu du répertoire principal de la disquette. Taper **c** : pour démonter la disquette et obtenir la liste de /

II. Vérifier si un utilisateur est connecté

III. Lire et traiter un fichier texte

- . Prérequis : while, les paramètres positionnels, set --

Conseil : utiliser la construction :

```
cat < users.txt | while true
do
  read ligne
  if [ "$ligne" = "" ]; then break; fi
  ...
done
```

- b. Créer un fichier texte **users.txt** contenant quelques lignes au format suivant login mot-de-passe nom groupes-secondaires
Par exemple : toto moiletoto M.Toto profs,reseau (pas d'espace dans les champs)
- c. Ecrire le script **essai-comptes.sh** qui parcourt ce fichier ligne par ligne, récupère les champs de chaque ligne dans les paramètres positionnels, et les affiche.
- d. Cet exercice sera poursuivi pour créer un script capable de générer des comptes à partir d'un fichier.

IV. Comptes créés

- Obtenir la liste de tous les utilisateurs (nom , uid, gid, répertoire personnel) possédant un compte créé sur le serveur, autrement dit ayant un uid supérieur à 500 (uid se trouve au 3ème champ de /etc/passwd)
- Se servir des méthodes vues dans l'exercice précédent avec les conseils suivants
 1. Extraire les champs 1,3 et 4 du fichier /etc/passwd avec la commande `cat /etc/passwd | cut -d: -f 1,3,4`
 2. envoyer les lignes précédents vers l'entrée de la commande tr, de façon à remplacer le séparateur ; par des espaces `cat /etc/passwd | cut -d: -f 1,3,4 | tr ":" " "`
 3. puis on envoie la sortie dans une boucle while qui permet d'en extraire chaque ligne, dont on affectera chaque champ aux paramètres positionnels grâce à **set --**
 4. Il suffira alors de comparer la valeur uid à 500 et d'afficher si uid >= 500

V. Tester un fichier

- Il s'agit de créer le script **test-fichier**, qui précidera le type du fichier passé en paramètre, ses permissions d'accès pour l'utilisateur
- Prérequis : passage de paramètres, instructions : test et if .. then .. else
- Appel : `./test-fichier nomFichier`
- Exemple de résultats attendus

Le fichier `/etc` est un répertoire

`"/etc"` est accessible par root en lecture écriture exécution

Le fichier `/etc/smb.conf` est un fichier ordinaire qui n'est pas vide

`"/etc/smb.conf"` est accessible par jean en lecture.

VI. Afficher le contenu d'un répertoire

- Prérequis : passage de paramètres, instructions : `for in`
- Appel : `./test-fichier nomFichier`
- Exemple de résultats attendus
- Prolongement difficile
Ecrire un script récursif capable d'afficher la liste des fichiers de l'arborescence dont la tête est passée en argument

VII. Envoyer un mail à un ensemble d'utilisateurs

Avec l'utilitaire **mail**, il s'agit d'envoyer un même message à un ensemble d'utilisateurs. On pourra écrire plusieurs versions :

- a. Le message est envoyé à tous les utilisateurs dont la liste est passée en argument au moment de l'appel (message1)
Indication : récupérer la liste des arguments dans la variable spéciale **\$@**
- b. Le message est envoyé à tous les utilisateurs actuellement connectés (message2)
Indication
Dans la boucle `for .. in liste`, on obtiendra `liste` avec une substitution de commandes, utilisant `who`
Amélioration
A la fin du script, tester si l'envoi s'est bien déroulé sans erreur en interrogeant le code de retour de **mail**. Si c'est le cas, ajouter à la fin du fichier `utilisateurs.send` : la date, et la liste des correspondants.
- c. Le message est envoyé à tous les utilisateurs dont le nom commence par un nom générique, passé en argument au moment de l'appel (message3)
Indication
On commencera par vérifier l'existence d'au moins un compte
l'appel **message stage** doit envoyer le message à tous les utilisateurs, dont le nom commence par `stage` (comme `stage1 ..`)
- d. Reprendre message1 avec le texte du message déjà enregistré dans le fichier `message.txt`
appel : `message message.txt liste`



TP1 scripts BASH : proposition de corrigés

I. Scripts a: et c:

```
#!/bin/bash
# script a:
echo --- taper c: pour démonter la disquette -----
mount /dev/fd0 /mnt/floppy
cd /mnt/floppy
ls
```

II. Un utilisateur est-il connecté ?

```
#!/bin/bash
# vérifier qu'un user est connecté
# appel : ./connexion.sh nom
if [ $# = 0 ]
then
echo "appel : $0 nom"
exit 1
fi
qui=$(who|cut -d" " -f1 | grep -w "^$1")
if [ $? = 0 ]
then
ou=$(echo $qui|cut -d" " -f2)
echo "L'utilisateur $qui est bien connecté(e) sur $ou"
else
echo "L'utilisateur $1 n'est pas connecté(e) actuellement"
fi
```

III. Lire un fichier

```
#!/bin/bash
# appel users.sh
# Lecture et affichage une par une des lignes de $fichier avec :
# cat < $fichier | read ligne
```

```
cat < users.txt | while true
do
    read ligne
    if [ "$ligne" = "" ]; then break; fi
    echo "lecture de la ligne --->" $ligne
    set -- $ligne
    login=$1
    passwd=$2
    nom=$3
    groupe=$4
    echo login=$login, mdp=$passwd, groupe=$groupe, nom=$nom
done
echo $?
```

IV. Trouver la liste des comptes créés

```
#!/bin/bash
# extraire la liste des comptes d'uid supérieur à 500
```

```
# appel : ./comptes.sh
```

```
echo Liste des comptes créés
echo =====
echo nom    uid    gid
```

```
# on peut aussi sauvegarder l'extraction dans un fichier temporaire passwd.txt
cat /etc/passwd | cut -d: -f1,3,4 | tr ":" " " | while true
do
read ligne
if [ "$ligne" = "" ]; then break; fi
set -- $ligne
if [ "$2" -ge 500 ]
then
echo $1 $2 $3
fi
done
```

V. Tester un fichier

```
#!/bin/bash
# syntaxe test-fichier  nomFichier
# effectue des tests sur le fichier et affiche un compte-rendu.
acces=""
if [ "$#" = 0 ]
then
echo "Syntaxe d'appel : $0 nomFichier"
exit 1
fi
if [ ! -e "$1" ]
then
echo "Le fichier \"$1\" n'existe pas !"
exit 1
fi
if [ -d "$1" ]
then
echo "Le fichier \"$1\" est un répertoire"
fi
if [ -f "$1" ]
then
echo -n "Le fichier \"$1\" est un fichier ordinaire"
    if [ -s "$1" ]
    then
echo " qui n'est pas vide"
    else
echo " qui est vide"
    fi
fi
if [ -r "$1" ]
then
acces="$acces lecture"
fi
if [ -w "$1" ]
then
acces="$acces écriture"
```

```

fi
if [ -x "$1" ]
then
acces="$acces exécution"
fi
echo "\"$1\" est accessible par $USER en $acces"
exit 0

```

VI. Lister un répertoire

```

#!/bin/bash
# syntaxe liste-fichier nom-fichier
# --> liste les fichiers comme la commande ls
if [ "$#" != 1 ]
then
echo "Syntaxe d'appel : $0 nomRepertoire"
exit 1
fi
if [ ! -d $1 ]
then
echo "\"$1\" n'est pas un nom de répertoire valide"
exit 1
fi
for fich in $1/*
do
if [ -d $fich ]
then
echo "d ---> $fich"
elif [ -f $fich ]
then
echo - $fich
else
echo "$fich : autre type de fichier"
fi
done

```

VII. Envoyer un mail à un ensemble d'utilisateurs

```

. script message1.sh
#!/bin/bash
# on pourrait utiliser une conditionnelle
# if [ $# = 0 ]; then
#   echo "Syntaxe $0 liste d'utilisateurs"
#   exit 1
# fi
# Attention: les ( ) sont obligatoires car && est plus prioritaire que ;

[ $# = 0 ] && (echo "Syntaxe $0 liste d'utilisateurs";exit 1)
echo "Envoi du message à $@"
for nom in $@
do
mail $nom@p00 <<EOF
bonjour a tous

```

```
ceci est un essai du script message1
utilisant mail sur p00
A +
Le "root"
EOF
done
```

b. script message2.sh

```
#!/bin/bash
# appel ./message2.sh
echo "message envoyé le $(date) a " >> utilisateurs.send
for nom in $(who | cut -d" " -f1)
do
mail $nom@p00 << EOF
bonjour a tous
Attention ! le root vous parle
déconnexion aujourd'hui a 18h
travaux de maintenance réseau ...
A +
Le "root"
EOF
echo "$nom " >> utilisateurs.send
done
```



TP2 scripts BASH

Objectifs

- I. Ecrire un script de création automatique d'un groupe d'utilisateurs
Puis le script de suppression de ce groupe
- II. Création de comptes Linux à partir d'un fichier.
Puis écrire le script de suppression des comptes décrit dans un fichier.
- III. Approcher la création automatique d'utilisateurs à partir d'une extraction de gep

Description par étapes

I. Créer un ensemble d'utilisateurs : `creer1.sh`

- Il s'agit de créer un ensemble de comptes constituant un nouveau groupe.
- Les noms doivent s'écrire comme un nom générique (par ex. stage, eleve ...) suivi d'un numéro.
- Le script demande d'abord le nom générique et celui du groupe secondaire dans lequel tous les comptes seront créés.
Par défaut le nom du groupe sera le nom générique.
- Détail :
 1. demander le nom générique et le nom du groupe
 2. tenter de créer le groupe. Si le groupe existe déjà (code de retour non nul) alors fin (exit 1).
 3. Ensuite on demande la saisie des numéros minimum et maximum.
 4. Demander la génération des comptes Samba (o/n)
 5. Créer les comptes par `useradd -G $groupe . . . .` dans une boucle while
Une variable numérique `$i` doit prendre toutes les valeurs de `$mini` à `$maxi` Indication : pour pouvoir incrémenter `$i` en fin de boucle, il faut la déclarer explicitement de type entier avec `declare -i i`
- Prolongement : faire générer des mots de passe standard Linux, puis Samba, formés des 3 premières lettres du nom de connexion suivi du numéro affecté à l'utilisateur.
Ajouter un compte de chaque compte créé avec la date dans un fichier `creer.txt`
- Voir un [corrigé](#)

II. Créer des comptes décrits dans un fichier : `creer2.sh`

- Récupérer l'éventuel argument passé sur la ligne de commande pour nom de fichier
S'il est absent, le demander à l'utilisateur, et par défaut de saisie, utiliser `user.txt`
- Vérifier l'existence de ce fichier : s'il n'existe pas, arrêter le script par une sortie du genre `exit 1`
- Lire le fichier ligne par ligne et traiter la ligne

III. Créer des comptes à partir d'une extraction GEP `creer3.sh`

- Récupérer une extraction de gep
 - Observer le séparateur de mots et la place des 3 champs login, mdp et groupe
 - Adapter en conséquence le script précédent
-

Annexes

1. La structure d'une ligne de `/etc/passwd` et de `/etc/group`

```
login:x:uid:gid:commentaire:home:shell
```

groupe:x:gid:liste-groupe-secondaires

2. Options de la commande **useradd** (pour détails cf man useradd)

useradd nom-login

- u uid (fixe l'uid)
- g groupe-primaire
- G liste de groupes secondaires (séparateur , sans espace)
- s shell (par défaut, attribution du shell par défaut bash)
- c commentaire
- d rep. personnel (home)
- e date d'expiration (format MM/JJ/AA)
- m recopie le contenu de /etc/skel dans le rep. home
- k rep-skel (sinon)

3. Options de la commande passwd (cryptage du mot de passe dans /etc/shadow)

passwd nom-login

- stdin** : la commande abandonne son caractère interactif habituel et examine son entrée standard pour s'en servir comme mot de passe (attention tout caractère est significatif, y compris les " ")
- d** : pour supprimer le mot de passe, l'utilisateur pourra se connecter sans !

4. Options de la commande smbpasswd

smbpasswd nom-login

- d** : permet de passer les mots de passe sur l'entrée standard (provenant d'un tube)



TP2 scripts BASH (corrigés)

1. Script de création automatique d'un groupe d'utilisateurs

pour un groupe d'utilisateurs, avec génération de mots de passe standard LINUX + SAMBA

```
#!/bin/bash
# SCRIPT creer1.sh
# =====
# création d'un ensemble de comptes construits
# avec un nom generique suivi d'un numero
# génération automatique de mot de passe Linux et Samba
# =====
echo "Donnez un nom de connexion générique"
read nom
echo "Donnez le nom du groupe (par défaut $nom)"
read groupe
if [ -z $groupe ]
then
groupe=$nom
fi

# le groupe existe t-il ? si oui, création impossible
echo "Création du groupe $groupe"
echo "===== "
groupadd $groupe
if [ $? -ne 0 ]
then
echo "Création du groupe impossible FIN"
exit 1
else
echo "Création du groupe $groupe " >>creer.txt
# pour trouver le numero gid du groupe, pas necessaire
# gr=`grep $groupe /etc/group | cut -d ":" -f3`
# echo "numero du groupe primaire $groupe : $gr "
fi

# saisie du mot de passe
# le mot de passe par défaut est fabriqué avec les 3 premières lettres
# suivies du numéro de l'utilisateur
motpasse=$(echo $nom | cut -c1-3)
echo "Donnez un mot de passe générique (par défaut $motpasse)"
read mdp
if [ -z $mdp ]
then
mdp=$motpasse
fi

# saisie des numeros
echo "numero mini"
read mini
echo "numero maxi"
read maxi
```

```

# création des comptes Samba ?
echo "Création (aussi) des comptes Samba ? (o/n)"
read rep
if [ $rep = "o" -o $rep="O" ]
then
smb="O"
fi

echo "Création des comptes $nom$mini a $nom$maxi"
echo "====="
# déclarer i variable de type entier (obligatoire)
declare -i i
i=$mini
while [ $i -le $maxi ]
do
  nomuser="$nom$i"
  motpasse="$mdp$i"
  # création du compte Linux
  useradd -G $groupe $nom$i
  echo $motpasse | (passwd --stdin $nomuser)

  # création du compte Samba avec le meme mot de passe
  if [ $smb = O ]
  then
    smbpasswd -a $nomuser $motpasse
  fi
  # affichage du compte-rendu
  uid=$(grep $nomuser /etc/passwd | cut -d: -f3)
  echo "compte Linux $nomuser ---> créé avec le numéro $uid et le mot de passe $motpasse"
  if [ $smb = O ]
  then
    echo "          compte SAMBA $nomuser ---> créé avec le mot de passe $motpasse"
  fi
  # ajout dans le fichier de log creer.txt
  echo "compte Linux $nomuser / uid = $uid / passwd = $motpasse, créé le $(date)" >>
  creer.txt
  if [ $smb = O ]
  then
    echo " & SAMBA $nomuser / $motpasse, créé le $(date)" >> creer.txt
  fi
  # au suivant !
  i=$((i+1))
done
echo "-----" >> creer.txt

```

2. Script de suppression du groupe

```

#!/bin/bash
# SCRIPT supprimer1.sh
# =====
# saisie du nom générique du groupe à supprimer
echo "Donnez le nom du groupe de comptes a supprimer"
read groupe
if [ -z $groupe ]
then

```

```

echo "Donnez un nom de groupe valide !"
exit 1
fi

# vérification : ce groupe existe t-il ?
# rappel : \b pour éviter que $groupe soit seulement le début du nom d'un groupe
if ! grep "^$groupe\b" /etc/group
then
    echo "ce groupe n'existe pas ... fin"
    exit 1
fi

echo "Suppression des comptes de $groupe"
echo "======"
liste=$(grep "^$groupe" /etc/passwd | cut -d ":" -f1)
echo "Liste des comptes à supprimer : "
echo $liste
# attention ! ne pas ecrire "$liste" ! pourquoi ??
set -- $liste
for nom in "$@"
do
    if userdel -r $nom
    then
        echo "suppression de $nom"
        # pour enregistrer les comptes effectivement supprimés dans un fichier
        echo "$nom est supprimé le $(date)" >>suppr.txt
    fi
done
# y a t-il encore un compte du groupe ?
if [ ! -e "/home/$nom"* ]
then
    echo "Le groupe est vide ; suppression du groupe (o/n) ?"
    read rep
    if [ $rep = "o" -o $rep = "O" ]
    then
        groupdel $groupe
        echo "Le groupe $groupe est supprimé le $(date)" >>suppr.txt
    fi
fi
echo "-----" >>suppr.txt

```

3. Créer des comptes décrits dans un fichier : creer2.sh

```

#!/bin/bash
# SCRIPT creer2.sh
# =====
# Création automatique des comptes d'après le fichier users.txt
# appel : ./creer2.sh [users.txt]

# analyse de la ligne de commande
# a t-on passé un paramètre ?
if [ $# = 1 ]
then
    fichier=$1
else
    echo "Nom du fichier des comptes Linux à créer [par défaut users.txt]"

```

```

    read fichier
    if [ -z $fichier ]
    then
        fichier="users.txt"
        echo "Utilisation du fichier $fichier pour genener les comptes"
    fi
fi
# le fichier $fichier existe t-il ? si non, sortie du script par exit 1
if [ ! -e $fichier ]
then
    echo "Le fichier $fichier n'existe pas ! Verifiez !"
    exit 1
fi

# Lecture une par une des lignes de $fichier
cat $fichier | while true
do
    read ligne
    # c'est la fin du fichier --> sortie brutale de la boucle
    if [ "$ligne" = "" ]
    then
        break
    fi
    # Traitement de la ligne (attention pas de " " autour de $ligne)
    # on "eclate" la ligne en affectant les variables de position
    set -- $ligne
    login=$1
    mdp=$2
    nom=$3
    groupe=$4
    echo $login $passwd $groupe $nom
    # existe t-il un sous-rep au nom $login dans /home/ ?
    if [ -d "/home/$login" ]
    then
        echo "Le compte $login existe deja ! "
    # on cree le compte avec $groupe comme groupe primaire
    else
        useradd -G $groupe $login
        echo $mdp | (passwd --stdin $login)
        smbpasswd -a $login $mdp
        uid=$(grep -w "^$login" /etc/passwd | cut -d: -f3)
        echo "creation de $login / uid = $uid / passwd = $mdp"
        echo "creation de $login / uid = $uid / passwd = $mdp, cree le $(date)"
>>creer.txt
    fi
done
# fin boucle de creation
echo "-----" >>creer.txt

```



Généralités sur les expressions rationnelles

Introduction

- *Le livre de référence utilisé : Introduction à PERL (O'Reilly)*
- Les expressions rationnelles (ou régulières) sont des critères ou modèles de recherche (pattern) dans les chaînes de caractères. Les objectifs peuvent être simplement de sélectionner suivant ce critère, ou d'effectuer des traitements comme des substitutions sur les chaînes trouvées.
- Leur utilisation s'étend de certains filtres shell : grep, sed, awk, vi, emacs .. à des langages de scripts : perl, php .. et aux éditeurs de texte : vi, emacs
- Attention ! certains caractères spéciaux sont communs avec les caractères génériques de désignation de fichiers, mais ils ont une interprétation différente. Donc il faut toujours prêter attention au contexte

Quelques exemples avec grep

- `grep abc fichier` recherche la chaîne **abc** dans toutes les lignes du fichier. Les lignes trouvées sont envoyées sur la sortie standard, éventuellement redirigée.
- `grep " " fichier` recherche les lignes qui contiennent un (et un seul) espace entre 2 mots
- `grep "ab*c" fichier`, idem avec 0 ou plusieurs occurrences de la lettre b
- `grep "^s.*n$" fichier`, reconnaît les lignes débutant par s (^s), finissant par n (n\$), avec éventuellement des caractères (quelconques) intermédiaires (.*)
- `grep "^([a-zA-Z][a-zA-Z0-9]*)" [a-f]*.txt` recherche les chaînes commençant par une lettre suivie d'un nombre qqc de caractères alphanumériques, dans tous les fichiers dont les noms débutent par une lettre de a à f avec une extension .txt

Quelques exemples en PERL

- Une expression rationnelle est une chaîne encadrée (en PERL) par `/.../`, appelé **opérateur de correspondance**, qui applique le modèle par défaut sur la variable `$_`, et renvoie vrai ou faux.
Par exemple, sur la ligne courante d'un fichier, contenue dans `$_`, recherche de la séquence **az** :

```
if (/az/) {
    print $_ ;
}
```

- Pour appliquer le modèle à une autre variable, il faut utiliser l'opérateur `=~` comme dans l'exemple :

```
if ($var =~ /az/) {
    print $var ;
}
```

- Exemples script `cherche-az`, pour rechercher la séquence "az":

```
# pour opérer sur toutes les lignes saisies au clavier ou récupérées sur l'entrée
standard
# rappel : chaque ligne de l'entrée est stockée dans l'argument par défaut $_
# appel : cat fichier |cherche-az
while () {
    if (/az/) { print $_ ; }
}
# pour opérer sur toutes les lignes d'un fichier
```

```
# appel recherche-az
open(IN, "~/essai.txt");
while () {
  chomp($_);
  if (/az/) { print "$_ \n"; }
}
```

Les méta-caractères

- Appelés aussi caractères spéciaux, ce sont des caractères interprétés *en contexte expression rationnelle* comme des opérateurs.
- En voici la liste avec un bref descriptif :

. (point) représente un caractère qcq, sauf \n
 * (astérisque) répétition du caractère précédent
 + au moins une occurrence de l'expression régulière
 ? au plus une occurrence de l'expression régulière
 [...] (crochets) l'un des caractères de l'ensemble.
 [^...] en début de crochets recherche dans le complémentaire de l'ensemble
 ^ recherche en début de ligne
 \$ recherche en fin de ligne
 \ annule le rôle de méta-caractère, pour jouer le rôle du caractère usuel
 {n,m} indique le nombre de répétitions attendus du caractère précédent
 | joue le rôle de "ou" entre 2 expr rég.

- L'antislash \ inhibe l'interprétation des caractères spéciaux et force leur interprétation usuelle.

Exemples

.\.**txt** recherche les chaînes du genre **c.txt** , où **c** est un caractère unique qcq
 ***\$** recherche les chaînes qui se terminent (\$) par le caractère astérisque (*)

Ecriture des motifs

Expression régulière à 1 caractère (atomique)

- un caractère correspond à lui-même, en règle générale.
Ainsi le motif **c** recherche le caractère désigné par **c**.
- un métacaractère précédé de \, pour lui rendre son rôle usuel.
- . Le point remplace tout caractère unique, sauf \n (newline)
 - **a.** toute suite de 2 caractères commençant par **a**, sauf **a\n**
 - **b.c** désigne toute suite de 3 caractères du genre **bac**, **bbc**, **bcc**, ...
- **[abc]** classe de caractères, sélectionne toute chaîne contenant l'un des caractères listés
 - **[a-z]** toute lettre minuscule
 - **[0-9]** équivaut à [0123456789], un chiffre quelconque
 - **[a-zA-Z0-9\ -_]** correspond à n'importe quelle lettre ou chiffre, ou au tiret ou au souligné
 -
- le caractère ^ juste après [joue le rôle d'exclusion des caractères qui suivent.
 - **[^aeiou]** tout **sauf** une lettre voyelle
 - **[^a-zA-Z0-9]** sélectionne un caractère non alphanumérique
- Certaines classes sont prédéfinies et servent d'abréviations

Construction	Classe équivalente	Construction de négation	Classe équivalente
--------------	--------------------	--------------------------	--------------------

\d (un chiffre)	[0-9]	\D (chiffres, non !)	[^0-9]
\w (1 caractère de mot)	[a-zA-Z0-9]	\W (mots , non!)	[^a-zA-Z0-9]
\s (espace)	[\r\n\f\t]	\S (space, non !)	[^\r\n\f\t]

Exemples

- [\da-fA-F] correspond à un chiffre hexadécimal quelconque
- 5[0-9][0-9] pour chercher les comptes d'UID 500 à 600.
- a[bc]d correspond aux chaînes abd ou acd.

Expressions régulières simples

Soit *expat* une *expression régulière atomique* quelconque, alors ce sont

expat1expat2expat3 ...	toute concaténation <u>sans espace</u> formée d'expr. régulière atomiques
expat*	chaîne composée de 0 à N caractères vérifiant expat
expat+	chaîne contenant <u>au moins</u> un caractère vérifiant expat
expat?	chaîne contenant <u>au plus</u> un caractère vérifiant expat
expat{n}	chaîne composée exactement de n caractères vérifiant expat
expat{n,}	chaîne composée d'au moins n caractères vérifiant expat
expat{n,m}	chaîne composée de n à m caractères vérifiant expat

Exemples

- **a*** caractère de multiplication, suivant un caractère signifie la répétition de 0 à plusieurs exemplaires de ce caractère. **[ab]*** signifie répétition possible d'un quelconque des 2 lettres a ou b.
- **a+** répétition de 1 à plusieurs exemplaires de a
- **a?** 0 ou 1 exemplaire de a
c[ad]?r impose la présence de cr ou car ou cdr, et rien de plus.
- **/x{5,10}** 5 à 10 répétitions attendues de x
- **a{5,}** 5 ou plus répétitions attendues de a
- **a{5}** exactement 5 occurrences de a
- **a.{5}b** recherche les mots contenant 5 caractères entre a et b
- **\s+** recherche un ou plusieurs caractères séparateurs.
- Conséquence : {0,} {1,} {0,1} correspondent à *, +, ?

Expressions régulières étendues

Soit *expsim* une *expression régulière simple* quelconque, comme définie précédemment, alors ce sont des expressions régulières générales :

expsim1expsim2 ...	toute concaténation formée d'expr. régulière simples
^expsim	recherche le motif <i>expsim</i> en début de chaîne

expsim\$	recherche le motif expsim en fin de chaîne
expsim\b	le motif expsim doit se trouver la fin d'un mot
expsim1 expsim2 ...	alternative : recherche de expsim1 ou expsim2 ...
(expsim)*	chaîne composée de 0 à N caractères vérifiant expsim
(expsim)+	chaîne contenant <u>au moins</u> un caractère vérifiant expsim
(expsim)?	chaîne contenant <u>au plus</u> un caractère vérifiant expsim
(expsim){n}	chaîne contenant la concaténation d'exactly n chaînes vérifiant expsim
(expsim){n,m}	chaîne composée de n à m caractères vérifiant expsim

Remarques

- Les symboles **^**, **\$** sont appelés *motifs d'ancrage*, car ils permettent de préciser l'endroit où doit être effectuée la recherche du motif dans la chaîne (alors que sans directive particulière la recherche de correspondance du motif avec la chaîne s'effectue de gauche à droite de la chaîne).
- Le symbole d'ancrage **\b** indique une limite de mot obligatoire, à l'endroit où il est inséré dans le motif
Par exemple :
`info\b` est vérifié par "l'info pour tous", "info-matin" mais pas par infos, informatique ...
`\binfo` est vérifié par "informatique", "m'informer" mais pas desinformation, ..
- Les parenthèses autour de **expsim** sont indispensables, sinon les opérateurs de répétition s'appliquent au dernier caractère de l'expression (il s'agit d'une question de priorité des opérateurs ..)
Par exemple, les motifs **li(nux)*** et **linux*** reconnaissent respectivement `li`, `linux`, `linuxnux`, .. et `linu`, `linux`, `linuxx`, `linuxxx`, ...
- Attention, ces extensions ne sont pas toutes reconnues par l'ensemble des filtres. Par exemple la commande `grep`, ne connaît pas `|`, ni les parenthèses

Exemples

- `color|couleur` réussit si l'un des motifs, soit `color`, soit `couleur`, est trouvé
- `5[0-9]{2}` reconnaît tout nombre de 500 à 599.
- `/^(eleve|prof|stage)[3-7]/` est satisfait par les chaînes commençant par l'un des 3 mots suivis immédiatement par un numéro de 3 à 7
- comment reconnaître un mot de 6 lettres formé des lettres (cela ne vous rappelle rien ?)

Parenthèses de mémorisation

La présence de parenthèses permet de mémoriser une ou plusieurs parties de l'expression qu'elle entoure, sans modifier son interprétation. Pour rappeler ces valeurs mémorisées, on utilise les notations `\1`, `\2` ... qui correspondent aux valeurs reconnues de même ordre.

Ainsi **eleve(\d)+poste\1** sera satisfait par *eleve3 au poste3* et non par *eleve5 au poste3*



Le langage PERL, résumé

Documentation et apprentissage

Avertissement : ces notes ont accompagné un auto-apprentissage de PERL. Elle n'ont donc aucune prétention pédagogique.

Le livre de référence abordable pour un débutant en PERL me semble être :

Introduction à PERL, par R.Schwartz & T.Christiansen (O'Reilly éd.)

La documentation complète est en cours de traduction en Français. Voir <http://www.enstimac.fr/Perl/>

Un cours de O.Aubert (Ecole Télécom Bretagne) à

<http://www.linux-kheops.com/doc/perl/perl-aubert/html/perl.html>

Caractéristiques générales

- Il s'agit d'un langage né vers 1986, dont la syntaxe générale, notamment les structures de programme, est proche du C. Il se veut proche du langage naturel, et notamment utilise beaucoup la notion de contexte.
- Le source est complètement analysé et compilé dans un format interne avant d'être exécuté (mais la forme compilée n'est pas conservée en standard)
- Les identificateurs commencent par une lettre ou un caractère souligné et peuvent aussi contenir des chiffres et des soulignés.
La casse est distinguée : jour, JOUR et Jour sont 3 identificateurs différents.
- Les fonctions prédéfinies peuvent au choix être appelées avec ou sans parenthèses `print "Bonjour"` , `"à tous !\n"; print ("Bonjour", "à tous !", "\n");`
- Appel d'un fichier exécutable contenant un source Perl.
En première ligne figure habituellement l'interpréteur du langage, le système insère alors le code nécessaire pour exécuter le script en tant que commande shell.

```
#!/usr/bin/perl -w
# Le modificateur -w facultatif permet de générer des diagnostics
# script hello.pl, à lancer par # ./hello.pl
# pour attribuer à tous le droit de l'exécuter : chmod +x hello.pl
print "Bonjour tout le monde\n";
```

- Ou appel directement en ligne de commande

```
perl -e 'print "Bonjour tout le monde\n";'
```

- Pas de déclaration, et les entrées-sorties sont très simplifiées

```
#!/usr/bin/perl -w
print "Quel est votre prénom ? ";
# <STDIN> symbolise l'entrée standard
$prenom = <STDIN>;
print "Bonjour $prenom, donnez un nombre : ";
# L'entrée standard peut être abrégée avec l'opérateur "diamant" <>
$nb = <>;
print " Vous avez choisi $nb \n";
```

Structures de données

Il y a 5 sortes de variables, dont le statut est différenciable grâce à un symbole préfixé différent.

type de variable	préfixe
scalaire	\$
tableau (liste)	@
associative (hachage)	%
fonction (routine))	&
fichier (handle)	<i>sans</i>

- Les variables scalaires sont simples, les autres sont des types structurés à l'aide de variables scalaires.
- Les listes (ou tableaux) sont indexés par des nombres, en commençant par l'indice 0.
- Les hachages (ou tableaux associatifs) sont indexés par des chaînes.
- Mais attention, comme tous les éléments des types structurés sont des scalaires, ils sont toujours préfixés par le symbole \$.

Variables scalaires

1. Identificateurs précédées du symbole \$.

Les variables scalaires peuvent être des nombres (entiers, flottants), des chaînes (ou des références de fichier). Les variables n'ont pas de type. Elles sont interprétées comme chaîne, nombre ou booléen suivant leur affectation ou le contexte. En général, la conversion entre chaîne et nombre est transparente.

Par exemple :

```
$a = '24' ; # $a est une chaîne
print $a + 1, "\n"; #affiche 25 ! attention pas de " "
```

Il n'y a pas de variable de type *booléen*. Mais en revanche une valeur scalaire peut être interprétée comme la valeur booléenne vraie (TRUE) si ce n'est pas une chaîne vide ou le nombre 0 ou la chaîne "0".

```
$x= 0.75;
$n= 123;
$y= 1.6e-19;
$octal= 015; #caractère retour-chariot
$hexa = 0xff; # vaut 255
$a=200; $b=350 ; $s= $a + $b;
```

2. Les chaînes sont délimitées soit par " " soit par ' '.

Dans le cas des guillemets, les variables incluses sont interprétées (on dit *interpolées* dans le jargon Unix) et remplacées par leur valeur.

```
$monNom ="Jean";
$salut = "Bonjour $monNom" ;
print $salut;    affiche bien Bonjour Jean !
$s = 'Bonjour $monNom !' ;
print $s;       affiche par contre Bonjour $s
```

```
$a=200; $b=350 ;
# affiche $a + $b = 200 + 350 = 550
print '$a + $b', " = $a + $b = ", $a + $b , "\n";
```

3. Utilisation de chaînes multilignes

On peut définir des chaînes sur plusieurs lignes

On utilise pour cela un identificateur (ici ESSAI) qui sert de délimiteur.

```
#!/usr/bin/perl
$texte = <<"ESSAI";
Bonjour a tous !
ceci est un essai
de texte multiligne
ESSAI
print $texte;
```

Ceci est très utilisé pour générer du code HTML à renvoyer au client Web

```
print <<"marqueur";
<body>
<h1>Voici un site WEB sur
<a href="http://www.ac-creteil.fr/infolyc/linux/formation">linux</a>
</h1>
</body>
marqueur
```

4. Variables spéciales prédéfinies

- **\$_**
Il s'agit d'un argument par défaut, déterminé par l'interpréteur en fonction du contexte.
En situation de lecture clavier ou fichier, **\$_** fait référence à la chaîne saisie.
En situation de recherche suivant un motif, **\$_** fera référence à ce motif.
- **\$\$**

5. La variable standard entrée

\$ligne=<STDIN> stipule de lire une ligne sur l'entrée standard jusqu'au caractère de validation \n, et l'affecte à la variable \$ligne, y compris \n

```
$ligne = <STDIN>;
chop($ligne);
print "La ligne saisie est : $ligne " ;
```

<STDIN> en l'absence d'affectation explicite, c'est la variable **\$_** qui reçoit la ligne entrée.

```
print "Pour sortir de la boucle Ctrl-C\n";
$i=0 ; # compteur de boucle
while <STDIN> {
  chop($_); i++ ;
  print "La ligne $i est : $_ \n" ;
}
```

6. Simulation de booléen

Il n'y a pas de variable de type booléen, mais on peut facilement utiliser à la place une variable numérique. En effet toute expression en situation de test (conditionnelle, boucle) valant 0 ou "0" ou undefined est considérée comme fausse, toute autre valeur est considéré comme vraie !

```
$trouve=0 # trouve est fausse
while (!$trouve) { # !$trouve = 1
  if (....) {
```

```

    trouve = 1 ; # trouve devient vraie
  }
}

```

Les listes ou tableaux

- Ce sont des ensembles ordonnés et indicés de scalaires. Les éléments d'une même liste peuvent être de types quelconques (nombre ou chaînes)
- Les identificateurs des listes sont précédés du symbole @, mais les éléments sont notés avec \$, car ce sont des valeurs scalaires
- Ils sont dynamiquement alloués, donc de dimension variable et leur nature peut être diverse : nombre, chaînes, variable, liste ...
- Pour créer une liste on peut :

- affecter directement une liste de valeur séparées par des virgules, en extension (énumérée) ou indiqué par des intervalles avec l'opérateur ..

```

@chiffre = (0,1,2,3,4,5,6,7,8,9);
@alphabet = (a..z, A..Z);

```

- inclure une liste dans une autre liste, qui peut être la même !

```

@alphanum = (a..z, A..Z , @chiffre);
@alphanum = (@alphabet , @chiffre);
@liste = (@liste , @chiffre);

```

○

- Variables listes prédéfinies

- **@ARGV** : arguments de la ligne de commande, numérotées à partir de 0 (\$ARGV[0]) est donc le 1er argument passé, et pas le nom du script qui est dans la variable **\$0**. (# shell !)
- **@INC** : chemins de recherche des fichiers requis par `require` ou `use`
- **@_** : liste contenant les paramètres des routines

- Pour connaître la liste

- Pour afficher l'ensemble des éléments : **print "@liste";**
- Pour connaître la taille : **\$taille = @liste;**
- Pour connaître l'indice du dernier élément : **\$i= \$#liste ;**

- Exemples de définition et de manipulation

```

@liste=(2,3,5,7);
print "le second élément est égal à ", $liste[1];

```

#la fonction qw() permet de simplifier la définition :

```

@mots=("voici", "mon","prénom", " : ", "Jean");
@mots= qw(voici mon prénom : Jean);

```

```

# ajout d'un 5ème élément au tableau @liste
$liste[4] = 9;

```

```

# affiche le dernier élément de la liste
print $liste[$#liste];

```

```

# longueur de la liste @liste

```

```

$lon = @liste;

# ajout de l'élément suivant du tableau @liste
$liste[@liste] = 11;

# affectation d'un tableau par une liste de type intervalle
@alphabet = ("a".."z");
@centaine =(0..99);
@dix-a-vingt = @centaine[9..19];

# équivaut à @indice = ("i".."k");
@indice = @alphabet[8..10];

# affectation d'une liste de variables scalaires par une liste
# on aura $var1="a"; $var2="b"
($var1, $var2) = @alphabet;

# affichage global de la liste (les " " servent à séparer les éléments
print "@liste";

parcours 1 de la liste avec foreach
foreach $element (@liste) {
print "$element  ";
}

parcours 2 de la liste avec for
# dans un contexte scalaire @liste est la longueur de la liste
# comme dans $lon = @liste;
for ($i =0 ; $i < @liste ; $i++ ) {
print "élément numéro $i ---> $liste[$i]\n";
}

```

Listes associatives (ou hachages, dictionnaires)

- Ce sont des tableaux indexés non par des indices entiers mais par des chaînes de caractères, appelées **clés**. Autrement dit, il s'agit d'un ensemble de couples (clé, valeur) dont le premier élément (clé) détermine le second (valeur).
- Les identificateurs des listes associatives sont précédés du symbole %, les éléments étant des scalaires
- Soit le hachage %tab et l'une de ses clés \$cle, alors sa valeur correspondante s'obtient par \$tab{\$cle} (attention avec des accolades).
Si la clé n'existe pas, on obtient la valeur **undef**
- Variables prédéfinies
 - %ENV : liste des variables d'environnement
 - %SIG : utilisé pour préciser les handlers
 - %INC : liste des fichiers qui ont été appelés par require
- Les opérateurs **keys()**, **values()**, **each()**
Ils s'appliquent à une liste associative et permettent respectivement d'extraire la liste des clés, la liste des valeurs, celle des couples (clé,valeur) en utilisant une boucle **while**.

```
@cles = keys(%tab);
@valeurs = values(%tab);
($cle, $val)= each(%tab);
```

- Exemples de définition et de manipulation

- %tab = ("pi" , 3.14 , "e" , 2.72, "q", 1.6e-19);
\$expo= "e";
print "La constante \$expo vaut : \$tab{\$expo}\n";
- %tab = ("jean" , 500, "toto" , 501, "stage1", 502);
print "La liste des clés : keys(%tab)\n";
print "La liste des valeurs : values(%tab)\n";
modèle de boucle pour parcourir le hachage
while ((\$cle, \$val)= **each** (%tab)) {
 print "Le numéro de \$cle est \$val\n";
}
même résultat, en plus compliqué ..
foreach \$cle (keys(%tab)) {
 print "Le numéro de \$cle est \$tab{\$cle}\n"
}

Opérateurs et expressions

Opérateurs scalaires

1. # de concaténation
print "salut"." jean"; *# affiche "salut jean"*
2. # puissance et affectation
\$x **=3; *# \$x=8*
3. # répétition d'une chaîne
print "jean " x 10; *#affiche "jean" 10 fois ...*
4. # affectations multiples
\$a = \$b = \$c = 5;
\$d = \$a +(\$e = 10); *# \$e = 10 et \$d = 15*
5. # opérateur incrémentation
\$a = \$b = 5;
\$c = ++\$a; *# \$c = \$a = 6*
\$d = \$b++; *# \$d = 5 et \$b = 6*
6. # les opérateurs **chop()** et **chomp()** chop retire le dernier caractère de la chaîne et retourne ce caractère
\$b= "Bonjour à tous";
\$a = chop(\$b); *\$b = "Bonjour à tou" et \$a ="s"*
chomp ne retire qu'un caractère nouvelle ligne "\n", sinon rien.
7. une instruction peut être suivi d'une expression modificatrice comme if, while ... Qq exemples /
 1. La fonction **defined** renvoie 1 (donc "vrai") si la variable qui suit est bien définie
print "Erreur, nom non défini !\n" if ! (defined \$nom);

Opérateurs et fonctions tableaux

On peut utiliser la structure tableau pour accéder différemment aux éléments de début ou de fin.

Comportement "pile" avec les fonctions push (empiler) et pop (dépiler)

```
@liste =(1..3);
push (@liste, 4 , 5);  # on peut empiler plusieurs valeurs, @liste=(1..5)
$val = pop (@liste); # donne $val=5 et @liste=(1,2,3,4)
```

Accès en début de liste avec les fonctions shift et unshift

```
@liste =(1..4);
unshift (@liste, 7); # insère $val en élément 0, donne @liste=(7,1,2,3,4)
$valeur = shift (@liste); # comme ($val, @liste)=@liste, donne $val=7 et
@liste=(1..4)
```

Structures de controle

Le vrai et le faux

En Perl, pas de type booléen. Toute condition est évaluée comme une chaine, les nombres sont ainsi convertis. Si la chaine résultante est soit vide "" ou bien "0", la condition est **false**. Dans tous les autres cas, elle est considérée comme **true**

Les opérateurs de comparaison

Opérateurs sur	Chaînes	Nombres
Égalité	eq	==
Différent	ne	!=
Inférieur	lt	<
Supérieur	gt	>
Inférieur ou égal	le	<=
Supérieur ou égal	ge	>=
Comparaison	cmp	<= >

Opérateurs booléens	
et	&&
ou	

L'opérateur de comparaison cmp pour les chaînes renvoie -1, 0 ou 1 selon que le premier argument est inférieur, égal ou supérieur au second.

instructions if et unless

Même syntaxe qu'en C ou en Java, à la différence que les accolades sont toujours **obligatoires**

Exemple

```
print "Quel est votre âge : ?"
$age = ;          # lecture console
chomp($age);
if ($age < 18 ) {
    # ou unless ($age >= 18 )
    print("Vous êtes trop jeune pour voter!\n");
} else {
    print("Vous pouvez voter!\n");
```

```
$inscrits++      # compte les inscrits au vote
}
```

Imbrication des conditionnelles

Elle est permise par la construction *if .. elsif .. else*

```
if (test1) {
    instruction1
} elsif (test2) {
    instruction2
} elsif (test3) {
    instruction3
} else {
    # les 3 tests ont échoué
    instruction3
}
```

instructions while et until

Syntaxe

```
while (condition) {
    bloc
}
until (condition) {
    bloc
}
```

Ces 2 boucles "tournent" tant que la condition demeure vraie (pour while) ou fausse (until).

Exemples d'utilisation

```
# boucle d'attente
while (<>) {
    last;
}
# lecture d'une ligne entrée au clavier, puis affichage
while ($ligne=) {
    print $ligne;
}

@nombres=(0..10);
print @liste,"\n";
print @lettres,"\n";print @nombres,"\n";$liste[5]=jean;print @liste,"\n";$i=0;
```

instruction for et foreach

- L'instruction **for** est identique à celle du C ou Java
-
- Exemple
@liste=(a..z);

```

lon = @liste;
for ($i=0 ; $i<lon ; i++) {
print "Mot numéro $i ---> $liste[$i] \n";
}
$i=0;
foreach $mot (@liste) {
print "Mot numéro $i ---> $mot \n";
$i++;
}

```

contrôle de boucle : last et next

Exemple

```

print "La demande sera arrêtée par la saisie de la lettre \"z\" ou \"Z\" \n";
while () {
chop($_);
if (($_ eq "z") || ($_ eq "Z")) { last; }
}

```

Les expressions régulières

Les modèles

- Ce sont des modèles ou formes à comparer aux chaînes de caractères (reconnaissance de formes). Elles sont très utilisées pour repérer des occurrences d'un modèle dans une chaîne à traiter, et permettre ainsi d'en réaliser des traitements comme des extractions de sous-chaînes.
- L'opérateur définissant un modèle en PERL est `/ ... /`. Par défaut, l'argument auquel s'applique l'opérateur `/` est la variable `$_`
- La chaîne ou ligne courante `$_` peut être remplacée par une variable qq,
même exemple en renommant \$ligne la ligne courante

```

if ($ligne =~ /html/) {
print $ligne;
}

# pour déceler la chaîne html dans la variable courante $_
if (/html/) {
print $_;
}

```
- Par exemple, le modèle `/html/` permet de repérer tout texte contenant la séquence *html*

Substitution de chaînes par `s///`

`s/expression-reg /nvelle valeur /options` Cet opérateur remplace les occurrences de l'expression dans la variable courante `$_`

```

if (/htm/) {
s/htm/html ;
print $_;
}

```

}

Pour une étude complète, voir perl-expr-regulieres.html

Entrées-sorties et manipulations de fichiers

Entrées-sorties standard

- Les accès aux fichiers s'effectue par l'intermédiaire de filehandles, variables particulières qui sont des descripteurs ou pointeurs de fichiers.
Perl dispose pour chaque script de 3 descripteurs prédéfinis STDIN, STDOUT et STDERR qui sont automatiquement initialisés (par le processus shell parent qui a lancé le script). Ces 3 variables sont donc immédiatement utilisables.
- L'opérateur de lecture noté **<STDIN>** lit normalement la prochaine ligne jusqu'au caractère nouvelle ligne "\n" (qui fait partie de la chaîne récupérée).

```
# pour placer la chaîne lue dans la variable $ligne
$ligne = <STDIN>
```

- Pour lire une à une les lignes, on utilise une construction while

```
while ($ligne=<STDIN>) {
  chomp($ligne) ; # pour enlever le caractère "fin de ligne"
  ..... # traitement sur $ligne
  print $ligne;
} # pour sortir de la boucle : ctrl-C
```

Entrées-sorties fichiers

- Pour se "connecter" à un fichier, il faut définir un *filehandle* qui pointe vers ce fichier (il est d'usage de choisir un identificateur en majuscules)

- Syntaxe

```
# ouverture en lecture du fichier
open(ENTREE, "<nomfichier"); # le caractère < est facultatif
# lecture d'une ligne du fichier
$ligne = (ENTREE, "<nomfichier");
le caractère < est facultatif
# création et écriture dans un fichier
open(SORTIE, ">nomfichier");
# ouverture en ajout dans un fichier existant
open(SORTIE, ">>nomfichier");
# en ajout dans un fichier existant
open(SORTIE, ">>nomfichier");
```

```
# fermeture
close(SORTIE);
```

- Exemples

Opérateurs de tests sur les fichiers

Les fonctions

- Syntaxe de déclaration

```
sub nom-fct {
  bloc instructions;
}
```

- Syntaxe d'appel

```
&nom-fct [liste-paramètres] ;
```

- Place et valeur de retour

- Comme le script est préalablement compilé, les définitions de fcts sont globales dans le script, leur place est indifférente et peuvent être appelées avant d'être déclarées.
- Les appels de fonctions peuvent être imbriqués sans limite, et bien peuvent être récursifs ! La valeur retournée est précédée de `return`, ou à défaut est la valeur renvoyée par la dernière instruction du corps de la fct.

- Variables locales et globales

Par défaut, les variables sont globales dans le script. Les variables locales doivent être explicitement déclarées avec les opérateurs **local()** ou **my()** avec la liste dans les ()

Par exemple, **local(\$p) = 1**; initialise à 1 la variable locale \$p1

Avec **my()** plus restrictive, la visibilité est limitée à la fonction courante, tandis que **local()** permet la visibilité dans les fonctions appelées par la fonction.

- Passage de paramètres

Lorsque l'appel de la fct est suivie d'une liste de paramètres (ou arguments) entre parenthèses, ceux-ci sont considérés comme des paramètres effectifs à transmettre à la fonction.

Les valeurs de ces arguments sont stockés par le compilateur dans la liste prédéfinie **@_**

Ces paramètres sont alors récupérables dans le code de la fonction sous les noms **\$_[0]**, **\$_[1]**, ...

- Exemple 1 calcul de la valeur absolue entre 2 nombres

```
#!/usr/bin/perl
print "Calcul de la distance entre \n";
print "x = "; $x=; chomp($x);
print "y = "; $y=; chomp($y);
# appel de la fct distance sans passage de paramètres
print "la distance vaut : ", &distance, "\n";

sub échange {
  # print "@\n";
  local($c)=0;    # $c variable locale d'échange
  $c = $_[0];     # $_[0] et $_[1] ont été affectés par $x et $y
  $_[0] = $_[1];
  $_[1] = $c;
}

sub distance {
  if ($x < $y) {   # $x et $y sont globales
    &échange($x, $y);
  }
  return $x - $y # retourne la distance
}
```

|}

- Exemple 2 calcul du produit d'une liste de nombres

```
#!/usr/bin/perl
# déclaration
sub produit {
    local($p) = 1;
    foreach $_ (@_) {
        $p *= $_;
    }
    $p; # pour retourner le résultat
}
# Utilisation
$p=3; # initialisation de la variable globale $p
print &produit(1,2,3,4), "\n"; # cet appel affiche 24
```



Les expressions rationnelles en PERL

Introduction

- *Le livre de référence utilisé : Introduction à PERL (O'Reilly)*
- Pour aborder les expressions rationnelles, voir [cette introduction](#)
- Les expressions rationnelles (ou régulières) sont des critères ou modèles de recherche (pattern) dans les chaînes de caractères. Les chaînes trouvées sont ensuite soumises à un traitement, en shell par des utilitaires comme grep, sed, awk, ou dans un langage de script comme Perl ...
- Attention !
Certains caractères spéciaux sont communs avec les caractères génériques de désignation de fichiers (comme * ?), mais ils ont une interprétation différente. Donc il faut toujours prêter attention au contexte

Quelques exemples élémentaires

- Une expression rationnelle est une chaîne encadrée (en PERL) par `/.../`, appelé **opérateur de correspondance**, qui applique le modèle par défaut sur la variable `$_`, et renvoie vrai ou faux.
Par exemple, sur la ligne courante d'un fichier, contenue dans `$_`, recherche de la séquence **az** :

```
if (/az/) {
print $_ ;
}
```

- Pour appliquer le modèle à une autre variable, il faut utiliser l'opérateur `=~` comme dans l'exemple :

```
if ($var =~ /az/) {
print $var ;
}
```

- Exemple : script `cherche-az`, pour rechercher la séquence "az":

```
# pour opérer sur toutes les lignes saisies au clavier ou récupérées sur l'entrée
standard
# rappel : chaque ligne de l'entrée est stockée dans l'argument par défaut $_
# appel : cat fichier |cherche-az
while () {
if (/az/) { print $_ ; }
}
# pour opérer sur toutes les lignes d'un fichier
# appel cherche-az
open(IN, "~/essai.txt");
while () {
chomp($_);
if (/az/) { print "$_ \n"; }
}
```

L'opérateur de correspondance

L'opérateur `/.../` et ses options

- Il s'agit de tester la présence dans une chaîne de caractères d'un motif particulier, et d'en trouver toutes les occurrences.
Pour cela on applique une expression régulière, soit **expreg** permettant de reconnaître le motif. En Perl, cette fonction

sera alors notée symboliquement **/expreg/**.

- On peut utiliser une autre syntaxe `m_expreg_` où les 2 caractères `_` doivent être remplacés par un même symbole délimiteur, par exemple :

```
if ($var =~ m%expreg%) {
print $var
}
```
- Ce symbole **m** dans l'expression `m/regexp/` est habituellement omis si le délimiteur de l'expression est un slash `/`. Dans ce cas, si un `/` est présent dans l'expression, il faut le "protéger" par `\/`
- Le motif **/expreg/** peut être suivi de paramètres dont voici le rôle :

g	la recherche est globale, de toutes les occurrences
i	ne pas distinguer minuscules et majuscules
s	traiter la chaîne comme une ligne simple (défaut)
m	traiter la chaîne comme une ligne multiple
o	ne compiler l'expression qu'une seule fois
x	utiliser les expressions régulières étendues

Interpolation de variables

Perl commence l'analyse de l'expression rationnelle en remplaçant les noms de variables par leur valeur

```
# recherche de "info" en début du texte
$mot="info";
$texte=<STDIN>; chomp($texte);
if ($texte =~ /^$mot/) { ...
```

Exemples

- Pour comparer le début de la saisie à la chaîne oui, quelle que soit la casse

```
if ( =~ /^oui/i) { ...
```
- Pour contrôler la présence d'un mot dans une chaîne

```
$inf="informatique";
$opt="option";
$chaîne="Toujours pas d'informatique enseigné comme option des lycées en l'an 2000
...";
if (($chaîne =~ /\b$inf\b/i) && ($chaîne =~ /\b$opt\b/i)) {
print "on parle enfin de l'option info !";
}
```

Valeurs de retour

La valeur retournée par la fonction dépend du contexte dans lequel elle est appelée.

Dans un contexte scalaire, la fonction renvoie une valeur non nulle en cas de succès, et sinon la valeur `undefined` `$code = ($chaîne =~ /expreg/);`

Dans un contexte de liste, la fonction renvoie la liste des éléments qui ont vérifié les expressions entre parenthèses. Si l'expression ne correspondait pas, on obtient une liste nulle.

```
($href) = ($chaîne =~ //i);
```

Dans tous les cas, la fonction fixera les variables 2, ... avec les éléments qui ont reconnu les expressions entre parenthèses.

Fonctions **s,split** et **join**

split décompose une chaîne en parties toutes séparées par un motif reconnu et construit une liste composée de ces éléments

Exemples

```
# @mots va contenir la liste des mots de la phrase, séparés par un ou plusieurs
espaces
$texte="Quel dommage que l'option informatique ait été supprimée !";
@mots= split(/ +/, $texte);

# dans la boucle, @liste va contenir la liste de tous les champs (même le second qui
est vide)
# une ligne étant de la forme jean:x:500:500::/home/jean:/bin/bash
$fichier="/etc/passwd";
open(F, $fichier);
while (<F>) {
@liste = split(/:/);
....
}
une meilleure solution pour un traitement sur les valeurs de la liste :
$fichier="/etc/passwd";
open(F, $fichier);
while ($user = <F>) {
($nom, $mdp, $uid, $gid, $titre, $home, $shell) = split(:/, $user);
....
}
```

Exercices

A suivre ...