Linux

Filesystem

Historique (Photo)

Le kernel et l’OS (Photo)

Filesystem, Filesystem

Le système de fichiers (filesystem) :

- Organisé hiérarchiquement dans sa totalité depuis la racine («/»)
- Sensible à la casse (des caractères)
- Utilise '/' pour séparer les répertoires dans un chemin
- Peut contenir n'importe quel caractère
- Taille pratique d'un nom de fichier illimitée
- Pas de notion «d'extensions»
- Distro-dépendant malgré le «standard»

Le système de fichiers (filesystem) (Photo)

L'organisation du filesystem (fs) : Une seule arborescence (Photo)

Filesystem, Balade dans le «fs»

● Se déplacer dans le filesystem :

cd chemin (chemin relatif)
cd /chemin (chemin absolu)

● Voir le contenu du répertoire :

ls (contenu du répertoire courant)
ls chemin (contenu de « chemin »)

● Connaître le répertoire courant : pwd (print working directory)

● Répertoires à part :
/ : racine du filesystem
. : répertoire courant
.. : répertoire parent
~ : répertoire maison («home dir», correspond à $HOME)

Filesystem, Lieux touristiques du «fs»

Répertoire essentiels au fonctionnement :

/ : racine
/bin, /sbin : binaires systèmes
/etc : configuration système
/boot : kernel et 2 eme étage du bootloader
/usr : binaires d'usage courant
/home : répertoires utilisateurs
/var : données variables (sgbd, logs)
/proc : information système temps réèl
/mnt, /media : point montage temporaire
/sys : bus systèmes
/tmp : répertoire temporaire
lost+found : objets trouvés

Commandes de base

Commandes de base, De l'aide : man

● man *commande*
affiche le manuel de la commande commande

– options
-t : produit une sortie postscript pour impression

– exemples :
man -t ls | lpr
man man

man divise la documentation en 9 sections :

1. Executable programs or shell commands
2. System calls (functions provided by the kernel)
3. Library calls (functions within program libraries)
4. Special files (usually found in /dev)
5. File formats and conventions (e.g. /etc/passwd)
6. Games
7. Miscellaneous (including macro packages and conventions
8. System administration commands (usually only for root)
9. Kernel routines [Non standard]

cela permet d'éviter les ambiguïtés :
man glob ≠ man 7 glob

Commandes de base, Aide : help, apropos, whatis

● help *commande*
affiche le manuel d'une commande interne (builtin)

● apropos *sujet*
affiche les pages de man correspondant au sujet

● **whatis *commande***
affiche une information succinte sur la commande

Commandes de base, cd

● **cd *argument***
« rentre » dans le répertoire argument

Après un changement de répertoire courant, l'ancien répertoire courant est stocké dans '-', permettant d'y revenir avec 'cd -'

– exemples
cd
(équivaut à 'cd ~' et 'cd $HOME')
cd ..
cd
cd /tmp
cd tmp

Commandes de base, $CDPATH

● $CDPATH permet d'offrir des chemins directement accessibles via « cd »
● Comme pour $PATH, les chemins sont concaténés avec ':' :

$ echo $CDPATH
/home/user:/var:/home/user/doc
$

Commandes de base, ls

● **ls -a**
Liste tous les fichiers (incluant .*)
● **ls -l**
Affichage long (droits, propriétaire, groupe, taille, date)
● **ls -R**
Liste les fichiers récursivement
● **ls -t**

Affiche les plus récents en premier

- **ls -S**

Affiche les plus gros en premier

- **ls -1**

Affiche le listing sur une colonne

Les options se combinent directement entre elles
ex : ls -laRt

- **ls *option argument***

liste les fichiers/répertoires correspondant à argument ou dans le répertoire argument

– options
-a : affiche tous les fichiers (y compris ceux commencant par '.')
-l : listing étendu
-R : récursif
-S : tri par taille
-t : tri par date de modification
-1 : affichage sur une colonne

– exemples

ls *.txt
ls /etc
ls /etc/host*
ls /etc/rc[13].d
ls /media/win/Program\ Files/
ls -laR~

Commandes de base, {mk,rm}dir

- **mkdir [-p] répertoire1 répertoire2 ...**

crée les répertoires répertoire1, répertoire2, ...
l'option -p per et destination (fichier ou répertoire)

– options
-p : crée les répertoires supérieurs si nécessaire

– exemples

mkdir $HOME/documents
mkdir test $HOME/images_iso
mkdir -p $HOME/documents/personnel/{photos,factures}

● rmdir répertoire1 répertoire2

supprime les répertoires répertoire1, répertoire2, ...
ces répertoires doivent être vides pour pouvoir être supprimés (utiliser rm rf sinon)
–
exemples

rmdir $HOME/documents
rmdir test $HOME/images_iso
rmdir $HOME/documents/personnel/{photos,factures,}

Commandes de base, cp & mv

● cp source destination

copie la source (fichier ou répertoire) vers la destination (fichier ou répertoire)

– options
-p : préserve les droits
-R : récursif
-f : force l'écrasement de la destination

– exemples

cp *.txt /tmp
cp test.txt toast.txt
cp Rf /home /var/backup

● mv source destination

déplace la source (fichier ou répertoire) vers la destination (fichier ou répertoire)
permet aussi de renommer un fichier

– exemples

mv *.txt /tmp
mv test.txt toast.txt

Commandes de base, rm & touch

● rm *argument*

Supprime le fichier ou répertoire argument

– options
-R : récursif
-f : force la suppression

– exemples
rm -rf /
rm toast.txt

● touch *fichier*
Crée le fichier s'il n'existe pas, ou met la date de modification du fichier à l'heure courante s'il existe
La commande ':>' permet aussi de créer un fichier

– exemple
touch toast.txt

Commandes de base, egrep

● egrep *options patron fichier*
liste les fichiers/répertoires contenant une chaine correspondant à patron.
si fichier n'est pas spécifié, grep travaille sur STDIN

– options
-v : inverse le comportement de egrep (n'affiche que les lignes qui ne correspondent pas)
-i : insensible à la casse
-R : récursif

– patrons d'expression régulières
si l'on utilise grep au lieu de egrep, il faut mettre un '\' devant les caractères ?, +, {, }, |, (, et)

. n'importe quel caractère
* le caractère précédent 0 ou plusieurs fois
+ le caractère précédent 1 fois au moins
? le caractère précédent 0 ou 1 fois
{n} le caractère précédent exactement n fois
{m,n} le caractère précédent de m à n fois
{n,} le caractère précédent n fois ou plus
[a - z] un caractère en minuscule
[a-zA-Z] une lettre
[0-9] un chiffre
^/$ le début/la fin de ligne
| séparateur pour spécifier de multiples expression (ou logique)

Commandes de base, egrep

● Matcher une ligne commençant par « foo » :
egrep "^foo.*"

● Matcher une ligne commençant par « foo » ou contenant « bar » :
egrep "^foo|bar"

● Matcher une ligne commençant par « foo » ou commençant par « bar » :
egrep "^(foo|bar)"

● Matcher une ligne commençant par « foo » et contenant « bar » :
egrep "^foo.*bar"

● Matcher une ligne commençant par « foo » et se terminant par « bar » :
egrep "^foo.*bar$"

● Matcher une ligne se terminant par un espace suivi d'un nombre de 1 à 3 chiffres :
egrep "[[:space:]]+[0-9]{1,3}$"

● Matcher une ligne se terminant par un espace suivi d'un nombre de 1 à 3 chiffres :
egrep "[[:space:]]+[0-9]{1,3}$"

● Matcher une ligne contenant le caractère '*' :
egrep "*"

● Matcher une ligne contenant le caractère '*' ou le caractère '?' :
egrep "*|?"

Commandes de base, cat, less, tee et wc

● cat *fichier1 fichier2 ...*
affiche le contenu de fichier1 fichier2 ... sur la sortie standard
si cat est appelé sans arguments, la source est l'entrée standard

– exemple
cat /dev/urandom

● less fichier1 fichier2 ...
comme cat, affiche le contenu de fichier1 fichier2 ... sur la sortie standard mais effectue un arrêt à chaque page
si less est appelé sans arguments, la source est l'entrée standard

– exemple
less /etc/password

● tee fichier

duplique l'entrée standard vers la sortie standard et dans un fichier

– exemple

vmstat 1 | tee toto

● wc option fichier

compte le nombre de lignes (-l), bytes (-c), mots (-w) .... dans fichier (ou sur STDIN si aucun fichier n'est spécifié)

- exemple

wc -l /etc/passwd

Commandes de base, tail et head

● head [-nX] *fichier1 fichier2 ...*

affiche les X dernières lignes de fichier1 fichier2 ... sur la sortie standard
si tail est appelé sans arguments, la source est l'entrée standard
– exemple :

head -n1 /etc/passwd

● tail [-nX] [-f] fichier1 fichier2 ...

affiche les X dernières lignes de fichier1 fichier2 ... sur la sortie standard
si tail est appelé sans arguments, la source est l'entrée standard et le nombre de lignes est 10
l'option -f permet de faire un 'tail' continu sur un fichier qui croît
– exemple

tail -n5 /var/log/syslog
tail -f /var/log/syslog

Une combinaison des deux permet d'afficher la nième ligne d'un fichier :
head -n10 /etc/passwd | tail -n1 : affiche la 9ème (10-1) ligne de /etc/passwd
head -n20 /etc/group | tail -n3 : affiche les lignes 17 à 20 (20-3 -> 20) lignes de /etc/group

Commandes de base, Versions ' z ' & ' bz '

● zcat *fichier1 fichier2 ...*
● bzcat *fichier1 fichier2 ...*

comme cat, mais sur des fichiers gzippés/bzippés.
Si zcat/bzcat est appelé sans arguments, la source est l'entrée standard
–exemples

zcat myfile.gz
bzcat myfile.bz2

- **zless** ***fichier1 fichier2 ...***
- **bzless** ***fichier1 fichier2 ...***

comme less, mais sur des fichiers gzippés/bzippés
si less est appelé sans arguments, la source est l'entrée standard
–exemples
bzless myfile.bz2

- **zgrep** ***options patron fichier***

duplique l'entrée standard vers la sortie standard et dans un fichier
–exemples
zgrep user myfile.gz

Shell : l’interpréteur de commandes

Shell, Shell

sh, sash, ksh, csh, tcsh, ash, bash, psh, fish,
...
On peut exécuter plusieurs shells en parallèle (unix est multitâche)
Plusieurs utilisateurs peuvent exécuter un shell en même temps (unix est multi-utilisateurs)

Autopsie d'un « login » (Photo)

Shell, Environnement

Un shell donné s'exécute dans un environnement propre et clos contenant des variables

Cet environnement :
– est modifiable
– est transitoire

L'environnement initial
– est défini au niveau système
– peut être modifié par l'utilisateur Shell

Shell, $PATH

● Variable d'environnement
● Contient une suite de chemins séparés par ':'
Les exécutables seront recherchés dans ces chemins

Shell, Répertoire '.'

● risque d'exécuter un trojan – danger ! n'importe qui peut écrire dans /tmp (entre autres) !
● le shell perd du temps à chaque 'cd'

Shell, Entrée/Sorties

● Par défaut tout processus possède :
– 1 entrée
– 2 sorties

● Ces I/O sont distinctes pour chaque processus

● Pour les programmes interactifs (comme les shells) :
– les entrées proviennent du clavier
– les sorties s'affichent sur l'écran

STDIN (entrée standard) : ce qui est envoyé vers le processus
STDOUT (sortie standard) : ce qui est envoyé par le processus
STDERR (sortie erreur standard) : les erreurs renvoyés par le processus

Ces entrés/sorties standard sont en fait des noms symboliques, correspondant à des « descripteurs de fichiers » :

● STDIN (Input) : descripteur de fichier 0
● STDOUT (OUTput) : descripteur de fichier 1
● STDERR (ERRor) : descripteur de fichier 2

Entrée/Sorties : cat (Photo)

cat : copie STDIN (ou un fichier) sur STDOUT (écrit les erreurs, s'il y a, sur STDERR)

user@host:~$ cat
azerty
azerty
qsdfg
qsdfg

user@host:~$ cat /etc/hosts

127.0.0.1 localhost
127.0.1.1 michel

The following lines are desirable for IPv6 capable hosts
::1
ip6localhost ip6loopback
fe00::0 ip6localnet

user@host:~$ cat /etc/bidon

cat: /etc/bidon: Aucun fichier ou répertoire de ce type

user@host:~$

Shell, Redirections d'E/S

Les E/S peuvent être redirigées de ou vers un fichier

processus < fichier (saisie utilisateur)
STDIN provient du fichier (et non plus du clavier)

processus > fichier (saisie utilisateur)
STDOUT est écrit dans fichier (et non plus sur le terminal)

processus 2> fichier (saisie utilisateur)
STDERR est écrit dans fichier (et non plus sur le terminal)

processus > fichier1 2> fichier2 (saisie utilisateur)
STDOUT est écrit dans fichier1 et STDERR dans fichier2

Redirections d'E/S (Photo)

user@pluton:~$ cat < /etc/hostname
pluton

user@pluton:~$ cat /etc/hostname
pluton

user@pluton:~$ cat /etc/hostname > /tmp/test

user@pluton:~$ cat /tmp/test
pluton

user@pluton:~$ cat /etc/hostname > /dev/null
user@pluton:~$ cat < /etc/hostname > /dev/null
user@pluton:~$ cat /etc/hostname
pluton

user@pluton:~$ cat /etc/portnaouak
cat: /etc/portnaouak: Aucun fichier ou répertoire de ce type
user@pluton:~$ cat /etc/portnaouak 2> /dev/null
user@pluton:~$ cat /etc/hostname /etc/portnaouak > out.txt 2> err.txt
user@pluton:~$ cat out.txt
pluton

user@pluton:~$ cat err.txt
cat: /etc/portnaouak: Aucun fichier ou répertoire de ce type

Shell, Redirections d'E/S (Photo)

user@pluton**:~$ cat /etc/hostname**
pluton
user@pluton**:~$ cat /etc/hostname >> /tmp/test**
user@pluton**:~$ cat /tmp/test**
pluton
pluton
user@pluton**:~$ cat /etc/hostname > /tmp/test**
user@pluton**:~$ cat /tmp/test**
pluton
user@pluton**:~$ cat /etc/hostname > /tmp/test**
user@pluton**:~$ cat /tmp/test**
pluton
user@pluton**:~$ cat /etc/hostname >> /tmp/test**
user@pluton**:~$ cat /tmp/test**
pluton
pluton
user@pluton:~$

Shell, Entrées/sorties : devices spéciaux

Plusieurs "devices" (fichiers dans /dev) ont une vocation particulière :

- **/dev/null**
trou noir annihilant tout ce qui lui est envoyé

- **/dev/zero**
envoie des zéros ad-vitam

- **/dev/random /dev/urandom**
fournisseurs officiels de hazard

- **/dev/full**
dispositif hypochondriaque : se plaint toujours (d'être plein)

Shell, Redirections d'E/S : travaux

1) Copier le contenu de /etc/passwd dans le fichier /tmp/users.txt
2) Ecrire «linus» à la fin de users.txt
3) Vider le fichier users.txt
4) Remplir users.txt de "zéros" (utiliser le dispositif fournisseur de zéros : /dev/zero)
5) Rediriger l'erreur standard de 'ls -lR /' dans /tmp/users.txt
6) Vider le fichier users.txt (d'une autre manière qu'en 3)

Shell, Pipes (Photo)

● Les «pipes» (pipelines) permettent d'envoyer la sortie d'une commande (STDOUT) à l'entrée d'une autre (STDIN) :

Shell, Pipes

● On trouve très souvent la commande grep au milieu de pipelines

● grep permet de n'afficher une ligne que si elle contient une chaine de caractères donnée

● Sa syntaxe est :

grep chaîne fichier
affiche les lignes de fichier contenant "chaîne"

grep chaîne
affiche les lignes lues sur l'entrée standard (STDIN) contenant "chaîne"

Shell, Pipes : exemples (compliqués)

user@pluton:~$ cat /etc/passwd | grep root
root:x:0:0:root:/root:/bin/bash

user@pluton:~$ ls | grep test
test.txt

user@pluton:~$ ip link | grep UP
1: lo: <LOOPBACK,UP,10000> mtu 16436 qdisc noqueue
3: eth1: <BROADCAST,MULTICAST,UP,10000> mtu 1500 qdisc pfifo_fast qlen 1000
6: tun0: <POINTOPOINT,MULTICAST,NOARP,UP,10000> mtu 1500 qdisc pfifo_fast qlen 100

user@pluton:~$ ip link | grep UP > uplinks.txt
user@pluton:~$ cat uplinks.txt | grep eth
3: eth1: <BROADCAST,MULTICAST,UP,10000> mtu 1500 qdisc pfifo_fast qlen 1000

user@pluton:~$ ip link | grep UP | grep eth
3: eth1: <BROADCAST,MULTICAST,UP,10000> mtu 1500 qdisc pfifo_fast qlen 1000

user@pluton:~$ history | awk '{ print $2 }' | sort | uniq c | sort nr k1 | head 10
164 ls
74 cd
62 ssh
55 ping
55 man
53 make
51 ip
47 more
user@pluton:~$

Il n'y a pas de limitation pratique au nombre de "pipes" que l'on peut enchaîner…

Shell, Pipes (Photo)

● Les pipes et les redirections peuvent être combinées

a 2> *dev/*null | b > */tmp/* fichier

Shell, Pipes & Redirections

Attention à l'ordre des redirections et des pipelines :

a | b 2> c

– le résultat de la commande a est passé à b
– les erreurs de b sont redirigées dans le fichier c

a 2> c | b

– le résultat de la commande a est passé à b
– les erreurs de a sont redirigées dans le fichier c

user@pluton:~$ cat /etc/toto /etc/passwd | grep root 2> /dev/null

cat: /etc/toto: Aucun fichier ou répertoire de ce type
root:x:0:0:root:/root:/bin/bash

user@pluton:~$ cat /etc/toto /etc/passwd root 2> /dev/null | grep root

root:x:0:0:root:/root:/bin/bash
user@pluton:~$

Shell, Complétion

● Permet de compléter une saisie utilisateur dans le shell
● Affecté à la touche « tab »
● La complétion affiche la plus grande correspondance unique
● S'applique aux commandes et à leurs arguments (selon configuration)
● « tab-tab » affiche toutes les correspondances possibles

Shell, Editeur de ligne

● l'éditeur de ligne permet d'éditer....
... la ligne !
● raccourcis clavier identiques à Emacs
CTRL + a : début de ligne
CTRL + e : fin de ligne
CTRL + k : coupe de la position courante jusqu'à la fin de la ligne
CTRL + y : colle ce qui a été précédemment coupé

Shell, Commandes multiples

● Le shell permet l'exécution de commandes successives en les séparant par ';' :
ls > ~/liste.txt; cd - ; ls >> ~/liste.txt

● Le shell permet d'exécuter une commande uniquement si la précédente a marché :
cat fichier > /dev/null && echo "le fichier existe"
grep-i "effacez moi" fichier && rm-f fichier

● Le shell permet d'exécuter une commande uniquement si la précédent a échoué :
cat fichier > /dev/null|| echo "le fichier n'existe pas"
grep i "gardez moi" ~/fichier.txt || rm f fichier.txt

Shell, Job control

●Fonctionnalité interne au shell
● Permet de gérer des tâches (processus) multiples dans un même shell :
– suspension temporaire : le processus est arrêté jusqu'à nouvel ordre
– arrêt définitif : le processus est terminé
– mise en arrière/avant plan : le processus reçoit l'entrée clavier
● Un processus peut ignorer l'arrêt définitif, mais pas la suspension

● Suspendre (stopper) un processus : <Ctrl>z
● Arrêter (terminer) un processus : <Ctrl>c
(si le processus est à l'avant plan)

● Arrêter (terminer) un processus : kill %n
(si le processus est à l'arrière plan)

● Voir la liste des processus du terminal (jobs) : jobs
'+' est le job « courant » (%% ou %+)
'-' est le job « précédent » (%-)

● Mettre un job en arrière plan : bg %n ou %n&

● Mettre un job en avant plan : fg %n ou %n

● Lancer un job en arrière plan : commande &

$ sleep 100
<ctrl-z>
[1]+ Stopped sleep 100
$ sleep 200 &
[2] 19858
$ jobs
[1]+ Stopped sleep 100
[2]-Running sleep 200 &
$ bg %1
[1]+ sleep 100
$ jobs
[1]- Running sleep 100
[2]+ Running sleep 200 &
$ kill %1
$ <return>
[1] – Complété sleep 100
$ fg
sleep 200 <ctrl-c>

$ jobs
$

Shell, Historique

● bash mémorise toutes les commandes dans un fichier d'historique
● cet historique est enregistré dans un fichier lorsque le shell se termine
● cet historique est lu depuis un fichier au démarrage d'un shell
● le nom du fichier est défini dans $HISTFILE et la taille de l'historique dans $HISTFILESIZE
le rappel de commandes :
– par numéro (!numéro)
– par nom (!debutnom)
– après recherche (ctrl-r)

user@host:~$ echo $HISTFILE
/home/user/.bash_history

user@host:~$ echo $HISTFILESIZE
1500

user@host:~$ history | tail -n 5
1429 history
1430 kill -9 18487
1431 echo $HISTFILE
1432 echo $HISTFILESIZE
1433 history | tail -n 5

user@host:~$ tail -n 5 $HISTFILE
ls
history
ls /ttest
history
kill -9 18487

user@host:~$!1431

user@host:~$ echo $HISTFILE
/home/user/.bash_history

user@host:~$!echo
echo $HISTFILE
/home/user/.bash_history
user@host:~$

user@host:~$ cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
...
postfix:x:111:118::/var/spool/postfix:/bin/false

user@host:~$!cat | grep root
root:x:0:0:root:/root:/bin/bash

user@host:~$ bash
user@host:~$ history | tail -n 5
1476 ls
1427 history
1428 ls /ttest
1429 history
1430 kill -9 18487
user@host:~$

Shell, Globbing

Permet de nommer plusieurs fichiers d'un seul coup grâce à des « jokers » (wildcards) :

● ? : accepte un seul caractère
● * : accepte n'importe quel caractère 0 ou n fois
● [chars] : dénote une liste de caractères acceptée (liste, suite alphabétique ou suite numérique)
● [^chars] ou [!chars] : dénote une classe de caractères refusée (liste, suite alphabétique ou suite numérique)
● {e1,e2,e3} : remplace les arguments par les éléments de la liste
● {e1,e2}{e3,e4} : remplace les arguments par le produit cartésien des deux listes
● abc? : accepte un fichier de 4 lettres commençant par abc
● abc* : accepte un fichier de 3 lettres ou plus commençant par abc
● *abc* : accepte un fichier de 3 lettres ou plus contenant abc
● [ab]c : accepte un fichier de 2 lettres commençant par 'a' ou 'b'
● [am]* : accepte un fichier commençant par une lettre comprise alphabétiquement entre 'a' et 'm' inclus
● *[0-1] [0-9] : accepte un fichier se terminant par un nombre compris entre '00' et '19'
● [13579]*[a-z] : accepte un fichier commençant par un nombre impair et se terminant par une minuscule
● [^13579]*[^a-z] : accepte un fichier commençant par un nombre pair et se terminant par une majuscule
● {a,b,c} : accepte les fichiers nommés 'a', 'b' ou 'c'
● {a*,*b*,*c} : accepte les fichiers dont le nom commence par 'a', contient 'b' ou se termine par 'c'
● *{.doc,.odt,.rtf} : accepte les fichiers nommés '*.doc', '*.odt' ou '*.rtf'
● {photo,image}{.jpg,.png} : accepte les fichiers nommés 'photo.jpg', 'photo.png', 'image.jpg' ou 'image.png'
● {jean,pierre}{,-jean} : accepte les fichiers dont le nom est 'jean', 'jean-jean', 'pierre' ou 'pierre-jean'

Shell, Globbing : classes

Les classes sont des listes prédéfinies de caractères, utilisables entre [: et :] :

● alnum : [a-zA-Z0-9]
● alpha : [a-zA-Z]
● ascii : caractère ascii (...)
● blank : espace ou tabulation
● cntrl : caractère de contrôle
● digit : [0-9]
● graph : caractères imprimables et visibles
● lower : [a-z]
● print : caractère imprimable
● punct : ponctuation
● space : espace, tabulation, ...
● upper : [A-Z]
● word : [a-zA-Z0-9_]
● xdigit : [a-fA-F0-9]

Exemple : [[:alnum:]] : accepte un caractère alphanumérique
Voir : man 7 glob

Shell, Globbing : les limites

- On ne peut pas matcher '/'
- On doit matcher explicitement le caractère '.'
- Ce ne sont pas des expressions régulières
- On ne peut donc pas appliquer de numération à des classes

Il est par exemple impossible de matcher tous les noms de fichiers se terminant par des nombres

Utilisateurs

Utilisateurs : /etc/passwd

● Utilisateurs définis dans le fichier /etc/passwd
– root (UID 0) : propriétaire de presque tous les fichiers système. Possède tous les droits
- les autres :
utilisateurs « système » (UID < 1000) : daemon, postfix, sshd, ...
vrais utilisateurs (UID ≥ 1000) : toto, marcel

● Mots de passe définis dans le fichier /etc/shadow. Seul root peut lire/modifier ce fichier

● Pour chaque utilisateur, le fichier /etc/passwd contient sept champs :

– login
– mot de passe ('x' pour les shadow passwords)
– UID (User ID)
– GID (Group ID, groupe principal)
– champ GECOS (nom complet, adresse, téléphones)
– répertoire personnel (« home dir »)
– le shell exécuté au login

Exemple :

root:x:0:0:Linux Torvalds,0,123,456:/root:/bin/bash

Utilisateurs : /etc/shadow

● Pour chaque utilisateur, le fichier /etc/shadow contient le mot de passe de connexion et ses paramètres de validité :
– login
– mot de passe chiffré
– 6 champs décrivant la validité du compte
– 1 champ réservé

Exemple :

mblanc:1QJ//btH...jL:13428:0:99999:7:::

Il n'est pas possible de se logguer directement si le mot de passe est '*' ou '!'

Utilisateurs : exemples

$ cat /etc/passwd

root:x:0:0:Linus Torvads:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
gdm:x:106:111:Gnome Display Manager:/var/lib/gdm:/bin/false
acox:x:1000:1000:Alan Cox,Kernel St,0625081221,0474701221:/home/acox:/bin/bash
$

$ cat /etc/shadow

root:*:13428:0:99999:7:::
daemon:*:13428:0:99999:7:::
bin:*:13428:0:99999:7:::
sys:*:13428:0:99999:7:::
sync:*:13428:0:99999:7:::
games:*:13428:0:99999:7:::
man:*:13428:0:99999:7:::
lp:*:13428:0:99999:7:::
mail:*:13428:0:99999:7:::
gdm:!:13428:0:99999:7:::
acox:1QN//abU4$nHskZjoAb3nx23J2z.WVJeqz.:13428:0:99999:7:::Utilisateurs

Utilisateurs, Groupes : /etc/group

● Chaque ligne contient 4 champs :
– le nom du groupe
– le mot de passe du groupe
– l'ID du groupe (GID)
– la liste des membres du groupe, séparés par des ','

● Chaque utilisateur possède en général un groupe à son nom (Unique Private Group), c'est son groupe primaire
● Chaque utilisateur peut aussi appartenir à n groupes secondaires
● Les groupes systèmes permettent souventaux utilisateurs de manipuler des devices (dialout, fax, audio, …)

Exemple :

$ cat /etc/group

```
root:x:0:
daemon:x:1:
bin:x:2:
sys:x:3:
adm:x:4:acox,ttso,ltorvalds
dialout:x:20:cupsys,acox,ttso,ltorvalds
fax:x:21:hugo,corentin
cdrom:x:24:haldaemon,acox,ttso,ltorvalds
floppy:x:25:haldaemon,acox,ttso,ltorvalds
tape:x:26:acox,ttso,ltorvalds
sudo:x:27:
audio:x:29:ttso,ltorvalds
wwwdata:x:33:
backup:x:34:
shadow:x:42:
utmp:x:43:
video:x:44:acox,ttso,ltorvalds
sasl:x:45:
plugdev:x:46:haldaemon,acox,ttso,ltorvalds
acox:x:1000:
ttso:x:1001:
ltorvalds:x:1002:
$
```

Utilisateurs, Création & gestion

● Outils « posix » :

{user,group}{add,mod,del}

● Outils « distro-dependant » :

{add,del}{user,group}

● Outils divers :

chsh, chfn

● Edition directe des fichiers
– déconseillée pour les créations/suppressions
– acceptable pour les modifications

Utilisateurs, Voir

●Qui suis-je ?
whoami, id, groups, who needs sleep

● Qui est là ?
who, users

● Qui était là ?
last, lastlog

● Qui fait quoi ?
w

● Qui existe ?
lastlog, cat /etc/passwd

Utilisateurs , Authentification : PAM

●L'authentification est gérée par PAM (Pluggable Authentication Modules)

● PAM permet de changer la façon dont on va identifier un utilisateur (globalement, ou pour un service donné)

● PAM gère 4 aspects de l'authentification :
– account : validité du compte, expiration du mot de passe, ...
– authentification : vérification de l'identité de l'utilisateur
– password : modification du mot de passe
– session : liste des tâches à effectuer avant la mise à disposition du service (ou après sa terminaison)

● PAM se configure dans
– /etc/pam.conf (configuration globale)
– /etc/pam.d/* (configurations spécifiques, par modules, …)

● Grâce à PAM, on peut authentifier un utilisateur de multiples façons :
- par son identifiant/mot de passe situés dans les fichiers passwd et group (méthode standard)
- par son identifiant/mot de passe stockés dans une base de données (MySQL par exemple)
- par un dispositif biométrique (empreintes digitales...)

- par un dispositif RFID/Bluetooth
- par une clef USB

...en résumé : l'authentification peut se faire :
● en fonction de n'importe quel(s) élément(s)
● service par service (par exemple Bluetooth pour login graphique, login/password pour la console)

Utilisateurs, Changer d'identité : su et sudo

● su («switch user») :
– permet d'exécuter une commande ou d'ouvrir un shell «en tant que»
– nécessite de taper le mot de passe de l'utilisateur désiré (sauf si l'on est root)

● sudo :
– permet d'exécuter une commande «en tant que»
– nécessite de taper SON mot de passe
– les possibilités sont définies dans /etc/sudoers

Droits

Droits, Généralités

● Chaque fichier ou répertoire possède :
– un propriétaire
- un groupe propriétaire

● Chaque fichier ou répertoire possède 3 listes de droits :
– les droits pour le propriétaire
- les droits pour le groupe
- les droits pour les autres

● Chaque liste donne les opérations possibles pour cet utilisateur :
– lecture (r) :
fichier : donne la possibilité de lire le contenu du fichier
répertoire : donne la possibilité de lire le contenu d'un répertoire (donc la liste des fichiers qu'il contient)
– écriture (w) :
fichier : permet d'écrire dans ce fichier
répertoire : permet d'y créer/renommer/supprimer des fichiers
– exécution (x) :
fichier : permet d'exécuter ce fichier
répertoire : permet d'y 'rentrer' (cd) et de voir son contenu

Droits, Fonctionnement (Photo)

● Type
b Block device
c Character device file
d Répertoire (Directory)
l Lien symbolique
s Socket
p FIFO
-Fichier normal

● Droit
r Read
w Write
x Execute (fichier)
x chdir (répertoire)
s SUID bit

● Destinataire

u User (propriétaire)
g Groupe
o Others (les autres, ni 'u' ni 'g')

Droits, chmod (Photo)

● chmod : permet de gérer les droits et peut fonctionner selon un mode littéral :

chmod *destinataire(s) opération droits, …*

- destinataire : u (user), g (group), o (other) ou a(all)
- opération : + (ajouter), - (supprimer), = (mettre à la valeur)
- droits : r (read), w (write), x (execute ou cd)

Exemples :

chmod ugo+rwx fichier1 fichier2
chmod u+rw, g+r, o-rwx fichier3
chmod ug=rwx,o=rx fichier4
chmod a+rx,u+w repertoire

● chmod : peut aussi fonctionner en mode octal :

– trois groupes d'utilisateurs (u, g, o)
– trois bits par groupe correspondant à r, w et x
– le mode s'exprime alors en octal

chmod 755 <=> chmod u=rwx,g=rx,o=rx

Droits, chown

chown : permet de changer le propriétaire ou le groupe propriétaire d'un fichier

chown *propriétaire:groupe fichier …*

● propriétaire : nouveau propriétaire du fichier ou répertoire
● groupe : nouveau groupe propriétaire du fichier ou répertoire
● fichier ... : fichiers ou répertoires dont il faut changer la propriété
● Si le groupe est omis, chown ne change que le propriétaire

chown root /etc/passwd

● Si propriétaire est omis, chown ne change que le groupe

chown :cdrom /dev/cdrom

Droits, umask

● Les droits d'un fichier ou répertoire à sa création sont défini par **umask**

● A sa création, un fichier aura les permissions :
666 - (umask)

● A sa création, un répertoire aura les permissions :
777 - (umask)

$ umask 0022
$ umask
0022
$ umask -S
u=rwx, g=rx, o=rx
$ touch fichier; mkdir repertoire; ls -ld fichier repertoire
-rw-r—r-- 1 tux zoo ... fichier
drwxr -xr-x 2 tux zoo ... repertoire
$

Droits, suid/sgid bits (très utile mais très dangereux)

suid/sgid bits (s) : celui qui exécute prend temporairement l'identité du propriétaire ou groupe propriétaire du fichier :

$ ls l zzz
rwxrxrx 1 ltorvalds ltorvalds 0 20070116 23:02 zzz

$ chmod u+s zzz
$ ls l zzz
rwsrxrx 1 ltorvalds ltorvalds 0 20070116 23:02 zzz

$ chmod g+s zzz
$ ls l zzz
rwsrsrx 1 ltorvalds ltorvalds 0 20070116 23:02 zzz
$

Droits, sticky bit

● sticky bit (t) : seul le propriétaire du fichier ou répertoire peut renommer ou supprimer un fichier dans un répertoire affligé du « sticky bit »

$ touch *fichier* **(crée le fichier *fichier)***
$ chmod a+rxw *fichier* **(ouvre tous les droits)**
$ chmod +t . *fichier* **(gèle les droits du répertoire courant)**
$ sudo chown root . *fichier* **(*root* devient propriétaire)**
$ ls -la **(affichage des droits)**

```
total 12
drwxrxrt    2    root        ltorvalds    4096    2007    01    17    00:17    .
drwxrxrx    3    ltorvalds   ltorvalds    4096    2007    01    16    23:02    ..
rwxrwxrwx   1    root        ltorvalds    0       2007    01    17    00:17    fichier
```

(*root* est devenue propriétaire de *fichier* créé par *Itorvalds*)

$ rm -f fichier
rm: ne peut enlever `fichier': Permission non accordée
$

Partitions & Filesystems

Filesystem , Types de filesystems

● minix
– fs original
– simple, encore utilisé sur les disquettes
– limité (64 Mb)

● ext2
– développé en 1993
– limité (2 Tb)

● ext3
– ext2 journalisé

● ext4
– supporte jusqu'au Pb (10 15)
– en cours d'intégration au kernel

● jfs/xfs
– journalisés
– respectivement IBM/SGI

● ReiserFS/Reiser4
– journalisé
– 10 fois plus performant sur les petits fichiers
– problème de pérennité ?

●iso9660/udf
- respectivement CDRoms/DVDRoms

● fat/vfat/ntfs
– fat/vfat bien gérés
– support ntfs plus délicat (ro)

● unionfs
– aggrégat de filesystems différents
– mélange possible entre ro/rw

● nfs/smbfs/coda
– filesystems réseau

ext3/ext4/jfs/xfs/ReiserFS/Reiser4 sont journalisés. Un filesystem journalisé note toutes les transactions à venir avant de les exécuter. En cas de crash, le système peut savoir ce qui a été fait et ce qui ne l'a pas été. La journalisation procure donc quelques avantages :

– la durée d'un fsck au boot ne dépend plus de la taille du filesystem (on sait ou chercher)
– intégrité des données accrue (et paramétrables)

Filesystem, Partitions (Photo)

● les disques sont découpés en partitions primaires (max : 4) et étendues
● les partitions sont représentées par des numéros à la fin du device représentant le disque
● les devices dépendent de l'interface du disque et de sa position (e.g. /dev/hda pour le 1 er disque IDE, /dev/hdb pour le 2 ème , etc.)

Filesystem, Filesystems (Photo)

chacune des partitions contient un filesystem («fs») qui sera monté dans l'arborescence du système

Filesystem, Devices et points de montage (Photo)

● Les disques et les partitions sont des devices
● Une partition contient la «racine» du fs
● Les autres partitions sont «montées» sur le fs

Filesystem Création de partitions (Photo)

sfdisk : ligne de commande (CLI)
fdisk, cfdisk : menus texte (curses)
gparted : interface graphique (gnome)

Filesystem Création de filesystems (Photo)

mkfs.<put_you_favorite_filesystem_here> <device>

user@host:~$ ls /sbin/mkfs.*

/sbin/mkfs.cramfs /sbin/mkfs.ext3
/sbin/mkfs.minix /sbin/mkfs.reiser4
/sbin/mkfs.vfat /sbin/mkfs.ext2
/sbin/mkfs.jfs /sbin/mkfs.msdos
/sbin/mkfs.reiserfs /sbin/mkfs.xfs
user@host**:~$**

● minix : fs linux historique, encore utilisé pour les disquettes
● cramfs : fs pour embarqué et initrd
● ext{2,3} : filesystem «standard» linux (ext3 journalisé)
● reiser4/jfs/xfs : reiserFS, IBM, SGI (jounalisés)
● msdos/vfat : dos (fat) & windows (fat-32)

Filesystem, Récréation : introduction à dd

dd est un cat sous stéroïdes :

● il peut contrôler le nombre d'octets écrits
● il peut convertir ces octets à la volée
● il peut lire/écrire des fichiers, des devices, des pipes, stdin/stdout...
● il peut sauter une partie de l'entrée ou de la sortie

dd if=/dev/zero of=mes_zeros count=1 bs=1024

bs : taille des blocs en octets (ici 1ko)
count : nombre de blocs
*la taille totale sera donc (bs*count)*

Filesystem, Création de filesystems

user@host:~/temp$ dd if=/dev/zero of=monfs.raw bs=1024 count=10240

10240+0 enregistrements lus
10240+0 enregistrements écrits
10485760 octets (10 MB) copiés, 0,08864 seconde, 118 MB/s

user@host:~/temp$ sudo losetup /dev/loop0 monfs.raw
user@host:~/temp$ sudo mkfs.ext2 /dev/loop0
mke2fs 1.39 (29May2006)
Étiquette de système de fichiers=
Type de système d'exploitation : Linux
Taille de bloc=1024 (log=0)
Taille de fragment=1024 (log=0)
2560 inodes, 10240 blocs
512 blocs (5.00%) réservé pour le super utilisateur
Premier bloc de données=1
Nombre maximum de blocs du système de fichiers=10485760
2 groupes de blocs
8192 blocs par groupe, 8192 fragments par groupe
1280 inodes par groupe
Superblocs de secours stockés sur les blocs :
8193
Écriture des tables d'inodes : complété
Écriture des superblocs et de l'information de comptabilité du système de fichiers : complété
Ce système de fichiers sera automatiquement vérifié tous les 23 montages ou
tous les 180 jours, selon la première éventualité. Utiliser tune2fs c ou i pour écraser la valeur.

user@host:~/temp$ sudo losetup d /dev/loop0

Filesystem, Montage de partitions

{u,}mount : {dé,}montage des partitions sur le filesystem :

mount t <fstype> <device> <mountpoint>
umount <device|mountpoint>

user@host:~/temp$ mkdir p /mnt/mountpoint/
user@host:~/temp$ sudo mount t ext2 o loop monfs.raw /mnt/mountpoint/
user@host:~/temp$ df /mnt/mountpoint/

Sys. de fich.	1Kblocs	Occupé	Disponible	Capacité	Monté sur
/home/leucos/temp/monfs.raw	9911	92	9307	1%	/mnt/mountpoint

user@host:~/temp$ sudo umount /mnt/mountpoint/

Filesystem, Partitions : /etc/fstab

Les partitions à monter au boot sont décrites dans /etc/fstab :

LABEL=ROOTFS
/
UUID=ed2d8d565c084bd4977e673f7a1966b2 /usr
/dev/hda1

(etc...)

Pour chaque filesystem à monter, /etc/fstab contient :

la partition à monter (/dev/hda1)
le moint de montage (/var)
le type de filesystem (reiserfs, ext3, ...)
les options de montage (lecteur seule, propriétaire, etc...)
un booléèn à 1 si le fs doit être sauvegardé par dump
un numéro d'ordre pour la vérification de fs au bootFilesystem

Exemple :

 # /etc/fstab: static file system information.
#
 # Use 'blkid' to print the universally unique identifier for a
 # device; this may be used with UUID= as a more robust way to name
#devices that works even if disks are added and removed. See fstab(5).
#
 # <file system> <mount point> <type> <options> <dump> <pass>
/dev/mapper/ubuntu --vg-root / ext4 errors=remount-ro 0 1
/dev/mapper/ubuntu --vg-swap_1 none swap sw 0 0

- Les partitions à monter sont désignées par :
– un device (/dev/hda1)
– un label (ROOTFS), créé par e2label ou tune2fs L
– un UUID, identifiant unique (ed2d8d56-5c08-4bd4-977e-673f7a1966b2), créé par tune2fs U

- Les UUID et labels permettent une indépendance par rapport au device

- **findfs permet de retrouver un fs par label ou UUIDFilesystem**

FIlesystem, Gestion

● df : donne l'occupation des filesystems montés :
df Th | grep "^/dev/"

● hdparm : tuning du disque (DMA, energie, ...) :
sudo hdparm /dev/sda

● tune2fs : tuning du filesystem (max-mount, UUID)
sudo tune2fs l /dev/sda2

● fsck : vérification du filesystem (boot single-user)
fsck a /dev/hda6Filesystem

Filesystem, Filesystems spéciaux

● procfs
– monté dans /proc
– contient des informations sur les processus
– zone fourre-tout pour les variables exportées du kernel

● sysfs
– monté dans /sys
– contient des informations sur les devices présents

Filesystem, Filesystems spéciaux : le swap

● L'espace de swap permet de décharger temporairement la mémoire physique (RAM)

● La taille empirique recommandée est 2 * RAM

● Lorsqu'un process n'utilise pas une zone de sa mémoire («page»), le kernel peut décider de la mettre dans le swap («swap out»)

● Lorsque la mémoire physique se fait rare, les zones nécessaires aux process :
– doivent être stockées sur le swap («swap out»)
– puis relues depuis le swap lorsque le process s'exécute («swap in»)

Filesystem, inodes & fichiers

● Dans un fs unix, chaque fichier est représenté par un inode

● Le fs contient une table d'association "nom de fichier" ⇔ "inode"

● L'inode contient toutes informations nécessaires concernant le fichier :

– numéro d'inode
– permissions, propriétaire
– dates (création, accès, modification)
– références vers les blocs de données

Filesystem, inodes & fichiers (Photo)

Filesystem, Liens symboliques : hard links (Photo)

● Un hard link est simplement une entrée de la DT qui pointe vers un inode

● Il peur y avoir plusieurs hard-links sur un inode (et donc plusieurs « noms » pour un même contenu)

● Les inodes existent tant qu'il y a au moins un hard-link pointant dessus

Filesystem, Liens symboliques : soft links (Photo)

● Les soft links (symbolic links / symlinks) pointent vers un autre « fichier » de la DT
● Le lien symbolique n'existe que par l'objet qu'il pointe : sans lui, plus de fichier

Filesystem, Hard links et soft links : choisir

● Utilisation des liens :
– hard links :
    ● ne peuvent fonctionner que sur un même filesystem (la tables des inodes est spécifique au filesystem)

    ● performances maximum (rien ne distingue un hard-link d'un autre)

- soft links :
    ● peuvent fonctionner entre les filesystems

- impact sur les performances (indirection)
- un symlink n'est rien sans son hard-link !

Filesystem, Liens symboliques : ln et stat

● ln permet de créér des liens

ln *original destination*

crée un hard link (destination) pointant sur l'inode de original

ln s original destination

crée un lien symbolique destination pointant sur original

● stat permet de connaître toutes les infos d'un inode

stat *fichier*

affiche le contenu de l'inode pointé par *fichier*

Filesystem, LVM : Logical Volume Management

● Fournit une abstraction du matériel de stockage :
– on travaille sur des volumes logiques et non plus sur des partitions
– on travaille sur des volume groups au lieu de disques
– on peut ajouter des partitions dans des volumes groups
– on peut agrandir les volumes logiques si nécessaire

● LVM permet au final de construire des systèmes de fichiers sur des devices ayant des tailles modulables

● Le système de fichier doit supporter le redimensionnement pour en profiter !

Filesystem, LVM (Photo)

Filesystem, RAID

● ***niveaux de RAID (linux-2.6.9+)***

– jbod (linear) : les disques sont concaténées
– raid0 (stripping) : un morceau de donnée sur chaque disque
– raid1 (mirroring) : un morceau de donnée sur tous les disques
– raid3 : la parité est stockée sur un disque
– raid5 : la parité est distribuée sur les disques de l'ensemble RAID
– raid6 : la parité est distribuée en double sur les disques de l'ensemble RAID
– raid 0+1 : un ensemble raid0 en miroir
– raid10 (1+0) : un ensemble raid1 strippé

● Gestion :
– /sbin/mdadm
– /etc/mdadm/mdadm.confFilesystem

Filesystem, Filesystems chiffrés

● CryptFS, EncFS, Loop-AES, losetup -e, TrueCrypt,
dm-crypt... à l'aide !

● Les choses se stabilisent, et deux méthodes émergent :
– TrueCrypt
    ● multi-plateforme (Linux/WindowsXP/200x)
    ● volumes chiffrés indétectables et indécelables

- Linux Unified Key Setup (luks)
    ●CryptoAPI kernel
    ● multiclefs
    ● volumes détectables

Filesystem, Editeurs

cat, echo >
toujours disponible
performances imbattables

vi
(presque) toujours disponible
très petit, incontournable, cryptique

CLI : aee, ed, e3, joe, jed, nano, pico,....
Gnome : gedit, gnotepad+, scite, scribes...
KDE : kate, kedit, …

Editeurs, Survivre à vi

● vi a deux modes :
  – mode commande : ce que l'on tape est considéré comme une commande et n'affiche rien
  – mode insertion : ce que l'on tape s'écrit dans le fichier

● Du mode commande au mode insertion : i, o, ...

● Du mode insertion au mode commande : <Esc>Editeurs

● Commandes vi :

n + ↵ : avancer de n lignes
i : passer en mode insertion
o : insérer une ligne en dessous et passer en mode insertion
dd : effacer la ligne courante
dw : effacer le mot courant
D : effacer la fin de la ligne courante
/ : rechercher
n : occurrence de recherche suivante
:w : écrit le fichier sur le disque
:q : quitte vi
:q! : quitte vi malgré les avertissements
:x : écrit le fichier sur le disque et quitte vi (équivaut à :wq)
n + G : aller à la ligne n. Si n est omis, va à la fin du fichier

Shell & commandes avancées

Shell & commandes avancées, Personnaliser le shell, Aliases

● Les alias permettent de créer une commande personnalisée

alias cd..="cd .."
alias la="ls la"
alias cat="echo Impossible de lire "

● ***alias*** sans arguments donne la liste des alias actuellement définis

● ***unalias*** permet de supprimer un alias
unalias cd..

Personnaliser le shell , ~/.bash{rc,_profile}

● L'environnement au démarrage du shell est paramétrable :

– ~/.bashrc : exécuté à chaque shell
– ~/.bash_profile : exécuté à chaque login shell

● Il est préférable d'utiliser ~/.bashrc, les login shell se font rares…

● La configuration par défaut d'un nouvel utilisateur est dans /etc/skel

● La configuration « system-wide » est définie dans /etc/bash.bashrc et /etc/profile

● Penser à personnaliser http_proxy

Commandes avancées, Boucles : for

● for/in : permet d'exécuter une série de commandes sur chacun des éléments d'une liste :

for *var* in *list* ; do *things* ; done

Exemples :

for i in 1 2 3 ; do echo i vaut $i ; done

i vaut 1
i vaut 2
i vaut 3

for i in mnt boot var; do echo Contenu de /$i; ls /$i; done

for i in b{in,oot}; do echo n "/$i prend "; du skh /$i; done

Commandes avancées, Boucles : backticks & seq

● seq : permet de générer une liste sur STDOUT

seq 1 9 ➔ affiche les chiffres de de 1 à 9
seq 1 2 9 ➔ affiche 1 3 5 7 9
seq 2 2 10 ➔ affiche 2 4 6 8 10

● Les `` (backticks) exécutent une commande et se remplacent par la sortie de cette commande :

rm `seq 1 3` : supprime les fichiers nommés 1 2 et 3

Commandes avancées, Boucles : for, `` et seq ensemble

for, `` et seq
for i in `seq 3`; do echo i vaut $i; done
for i in `seq 1 2 9`; do echo i vaut $i; done
for i in `seq 9 2 1`; do echo i vaut $i; done
for i in `ls ~`; do echo Il y a $i dans $HOME; done
for i in ~/tmp/*; do echo n Le fichier $i est de type" "
file b $i
done
for i in ~/tmp/*; do echo n Le fichier $i est de type" "
file b "$i"
done
for i in ~/tmp/*; do echo n Le fichier \'$i\' est de type" "
file b "$i"
done

Commandes avancées, Traiter le texte : cut

cut : permet d'extraire une colonne dans un fichier à champs délimités

cut f<i> d<délim> <file>

– cut lit sur STDIN si file est omis
– le séparateur de champ est délim (blanc si omis)
– extrait le i ème champ sur chaque ligne et affiche sur STDOUT

cut -f1 -d':' /etc/passwd ➔ affiche tous les utilisateurs déclarés dans /etc/passwd

Commandes avancées, Traiter le texte : sort

sort : permet de trier les lignes lues dans un fichier (éventuellement à champs délimités)

sort k<k> [n] [r] t<délim> <file>

– sort lit sur STDIN si file est omis
– le séparateur de champ est délim (ou du blanc si délim est omis)
– compare en fonction de la clef à la position k
– avec l'option -r, l'ordre est renversé (z->a, 9->0)
– avec l'option -n, la comparaison n'est plus alphabétique mais numérique

sort –k*k* -n -t ':' /etc/passwd ➔ affiche /etc/passwd dans l'ordre numérique des UID

exemple : sort -k1 -t’ :’ etc/passwd

Commandes avancées , Traiter le texte : uniq

uniq : permet de supprimer les lignes consécutives doublon lues dans un fichier

uniq [c] <file>

– sort lit sur STDIN si file est omis
– n'affiche qu'une ligne lorsque plusieurs lignes consécutives sont lues
– avec l'option -c, uniq affiche le nombre d'occurrences pour chaque ligne

echo e "aaa\naaa\naaa\nbbb\nbbb" | uniq ➔ n'affiche que 'aaa' et 'bbb'

(pour la commande ‘echo’, l’option ‘-e’ active l’interprétation de ‘\’ comme n’étant pas un caractère ; ‘ \n ‘ est interprété par ‘echo’ comme ‘ligne nouvelle’)

Commandes avancées, (Mal)traiter le texte : sed

sed : remplacement de texte ("stream editor")

sed [-r] 's/motif1/motif2/' [fichier]

– lit fichier ou STDIN si aucun fichier n'est spécifié
– -r permet d'utiliser des expressions régulières (presque) compatibles avec egrep
– remplace 'motif1' par 'motif2' (les motifs sont des expressions régulières compatibles avec grep)
– \1...\n dans motif2 sont des variables positionnelles représentant un match dans motif1
– écrit le résultat sur STDOUT

echo "chat noir" | sed -r 's/noir/blanc/' ➔ affiche "chat blanc"

● le modificateur ' g ' permet d'attraper plusieurs fois le motif sur la même ligne :

echo "chat noir, très noir" | sed r 's/noir/blanc/' ➔ affiche « chat blanc, très noir»
echo "chat noir, très noir" | sed r 's/noir/blanc/g ➔ affiche « chat blanc, très blanc»

● le modificateur ' i ' permet d'ignorer la casse lors des évaluations :

echo "chat NOIR, très noir" | sed r 's/noir/blanc/g' ➔ affiche « chat NOIR, très blanc»
echo "chat NOIR, très noir" | sed r 's/noir/blanc/gi' ➔ affiche « chat blanc, très blanc»

– \1...\n dans motif2 sont des variables positionnelles
– ces variables référenceront les expression entre () dans motif1
– elles ne sont initialisées que s'il y a une concordance

echo e "chat noir" | sed r 's/chat (.*)/\1 chat/' ➔ affiche "noir chat"
echo e "abc" | sed r 's/(.)(.)(.)/\3\2\1/' ➔ affiche "cba"

- Rappel sur les expressions régulières
. n'importe quel caractère

* le caractère précédent 0 ou plusieurs fois
\+ le caractère précédent 1 fois au moins
? le caractère précédent 0 ou 1 fois
{n} le caractère précédent exactement n fois
{m,n} le caractère précédent de m à n fois
{n,} le caractère précédent n fois ou plus
[az] un caractère en minuscule
[azAZ] une lettre
[09] un chiffre
[[:alnum:]]: [azAZ09]
^/$ le début/la fin de ligne
[[:alpha:]]: [azAZ]
| séparateur pour spécifier de multiples expressions (ou logique)
[[:ascii:]]: caractère ascii (...)
[[:blank:]]: espace ou tabulation
[[:cntrl:]]: caractère de contrôle
[[:digit:]]: [09]
[[:graph:]]: caractères imprimables et visibles

– Classes de caractères
[[:lower:]]: [az]
[[:print:]]: caractère imprimable
[[:punct:]]: ponctuation
[[:space:]]: espace, tabulation, ...
[[:upper:]]: [AZ]
[[:word:]] : [azAZ09_]
[[:xdigit:]] : [afAF09]

Commandes avancées, Rechercher des fichiers : find

find *argument options*

liste les fichiers/répertoires situés dans argument selon des critères définis dans *options*.

– options
-name glob : liste les fichiers correspondant au shell glob glob
-type f/d/l : ne liste respectivement que les fichiers, répertoires, liens
-exec : exécute une commande récursive

– patrons de globs
. n'importe quel caractère

* e caractère précédent 0 ou n fois

\+ le caractère précédent 1 ou n fois

? le caractère précédent 0 ou 1 fois

[az] un caractère en minuscule

[azAZ] une lettre

[09] un chiffre

exemples :

find ~/mes_docs type f name "*.doc"
find ~/mes_docs type f name "*facture*" exec rm f '{}' \;
find / name "*log*" type d

Commandes avancées, xargs , companion de find

● xargs permet d'utiliser la sortie d'une commande et de la passer en argument à une autre

cut f6 d':' /etc/passwd | xargs du skh

● La commande xargs est souvent utilisée en combinaison avec find :

find arguments | xargs commande

● C'est l'équivalent de 'find -exec', la performance en plus :
– find -exec exécute un process pour chaque résultat
– xargs cumule la sortie pour maximiser le passage d'arguments

Commandes avancées, Rechercher des fichiers : locate

● find effectue une recherche sur le filesystem en temps réel : long

● Pour la recherche par nom de fichier, locate permet de faire la même chose

● locate utilise une base de données pré- générée par updatedb

● locate permet de rechercher des expressions régulières (find ne peut rechercher que des globs)

Shell scripting, Bases

● Un shell script est un fichier exécutable contenant une série de commandes shell

● Un script commence généralement par : #!/bin/sh

● Il doit être exécutable, ou appelé par 'sh script'

● Par convention, il se termine en général par « .sh »

● Les lignes commençant par '#' sont des commentaires

Shell scripting, Variables

● Dans un script, on peut utiliser des variables locales ou issues de l'environnement :

echo $HOME

● En dehors de l'affectation, la variable doit être précédée par '$'

root@jeanpierre:~# mavar=bonjour
root@jeanpierre:~# echo $mavar
root@jeanpierre:~# bonjour

● Les arguments passés aux scripts sont automatiquement affectés aux variables $1, $2, …

● La variable $? donne la valeur de sortie numérique de la dernière commande

● Un résultat nul indique (en général !) que la commande a réussi

user@host:~$ touch f
user@host:~$ cat f
user@host:~$ echo $?
0

user@host:~$ rm f
user@host:~$ cat f
cat: f: Aucun fichier ou répertoire de ce type

user@host:~$ echo $?
1

Shell scripting, Branchements

● La construction if/then/else/fi permet d'effectuer des branchements conditionnels en fonction de valeur de sortie de commandes :

echo n 192.168.0.254 est
if ping c1 192.168.0.254
then
echo joignable
else
echo injoignable
fi

● La construction if/then/else/fi permet d'effectuer des branchements conditionnels en fonction de valeur de variables :

if ["$USER" = "root"]
then
echo Bonjour maitre
else
echo Encore un utilisateur de base...
fi

● La construction [ ] est l'équivalent de la commande 'test'

Shell scripting, Comparaisons

● La commande test (ou '[]') permet de comparer des valeurs entre-elles

– Comparaison numériques :
-eq ne gt ge lt le
égal (equal) différent (not equal) plus grand que (greater than) plus grand que ou égal à (greater or equal) plus petit que (less than) plus petit que ou égal à (less of equal)

-Comparaison de chaînes :
– = != < >
égal (chaînes identiques) différent précède (tri alphabétique) suit (tri alphabétique)

Shell scripting, Arithmétique

● Le shell offre trois possibilités pour formuler des calculs :

– la commande expr expression :
valeur=`expr 10 \* 20`
valeur=`expr $valeur + 2`

– la construction $[expression]
valeur=$[10 * 20]
valeur=$[$valeur + 2]

– la construction $((expression))
valeur=$((10*20))
valeur=$(($valeur+2))

Shell scripting, Boucles

● while condition/do/done permet de boucler tant que la condition est vraie

num=1
while [$num le 5]; do
echo $num
num=$[$num + 1]
done
while ping c1 $host > /dev/null 2>&1; do
echo $host est joignable
done
echo $host est Injoignable

Gérer et superviser les ressources

Gérer les resssources, Gérer le temps

● Mettre le système à l'heure
– Depuis un serveur ntp :

ntpdate time.erasme.org

– A la main :

date set="20061224 23:59:00"

● Lire l'heure :
date +FORMAT

ex : date +%Y%m%d" "%H:%M:%S

● Chronométrer

time ***commande***

Gérer les ressources, Gérer les processus

● Chaque exécutable, script, démon, apparaît comme un processus sur le système lorsqu'il est lancé

● Lorsqu'un processus est lancé, le kernel lui attribue un numéro (**PID**) La commande ***jobs*** ne permet de voir que les processus exécutés dans le terminal courant

● Pour visualiser les autres processus, il faut utiliser 'ps'

Gérer les ressources, Gérer les processus : ps

●Voir tous les processus :

ps edf

Voir tous les processus, un peu plus de détail :

ps awux

Voir les processus de l'utilisateur courant :

ps wux

Gérer les ressources, Gérer les processus : kill{,all}
● kill permet d'envoyer in signal à un processus
● ce processus peut intercepter ce signal et agir en conséquence
●kill n'est pas forcément malfaisant !
●man kill donne la liste de signaux possibles
●killall permet d'envoyer un signal à des processus par leur nom
●un utilisateur ne peut signaler que ses processus
●attention : kill est souvent un built-in (et non /bin/kill), donc help kill et non man kill pour de l'aide !

kill signal pid
killall signal nom

– pid : numéro du process
– signal : numéro ou nom du signal (1 envoie le signal à tous les processus)

STOP/CONT : équivalents à <Ctrl>-z et fg/bg pour les process du terminal
INT (2) : arrêt demandé par l'utilisateur (généralement via <Ctrl-C>)
TERM (15) : kill demande gentiment au process de se terminer
KILL (9) : le processus est tué sans sommation
HUP (1) : signal de déconnexion du terminal, maintenant surtout utilisé pour demander une reconfiguration
USR1 : généralement utilisé pour demander une reconfiguration

Gérer les ressources, Quotas disque

La suite d'outils 'quota' permet de limiter l'espace disque et le nombre d'inodes :
– par usager
– par groupe
Ces limites sont soit :
– rigides : dès qu'elles sont atteintes, impossible de créer d'autres fichiers
– souples : lorsque les limites sont atteintes, l'utilisateur peut encore consommer de l'espace pendant une période de « grâce »

Gérer les ressources, Limites « PAM »

● PAM permet d'appliquer un certain nombre de limites à un utilisateur ou à un groupe

● Ces limites sont définies dans /etc/security/limits.conf

● Elles sont appliquées à la catégorie « session »
– nombre max de process

– mémoire maximum
– priorité des processus
– ...
➔ voir ulimit a

Superviser le système

Superviser le système, Problèmes classiques

● Filesystem plein
– 90% des problèmes spontanés
– résolu facilement en augmentant la capacité du fs
➔ installez LVM

● Filesystem corrompu
– pertes très rares avec les fs journalisés
– résolu facilement avec une politique de backup efficace
➔ installez bontmia/dump/arkeia/…

Crash disque
– Un setup RAID (soft ou hard) permet de se prémunir facilement d'un crash disque
– Un reboot sera peut-être nécessaire en fonction du bus du disque
➔ utilisez le RAID (hard ou fourni par le kernel)

Superviser le système, Problèmes classiques

● Surcharge système
– traquer les gourmands
- optimiser le système
➔ utilisez ps/top/vmstat/sar...

● Facteur humain
– de loin, la plus forte cause d'indisponibilité du système
- résolu laborieusement par de la pratique et la destruction de nombreux systèmes innocents (de production de préférence)

● Voir tous les fichiers ouverts
lsof

● Voir tous les processus qui ouvrent fichier
lsof fichier

● Voir tous les processus qui ouvrent un ***fichier*** sous ***répertoire***
lsof répertoire

● Voir tous les fichiers ouverts par le processus PID
lsof pPID

● Afficher en boucle toutes les S secondes
lsof rS

● Voir tous les processus ayant une connexion ouverte avec host
lsof i @host

● Voir tous les process ayant une connexion ***tcp*** sur le port ***port***
lsof i TCP:port

● Voir tous les processus ouvrant des fichiers sur un device ou point de montage
fuser vm <device|mountpoint>

● Idem, en effectuant le « ménage »
fuser kvm <device|mountpoint>

Superviser le système, Gérer l'espace disque

● Voir l'occupation de tous les filesystems
df Th

● Voir l'occupation du filesystem contenant le répertoire rep
df Th rep

● Voir l'occupation de tous les filesystems ext3
df ht ext3

● Voir l'occupation de chaque sous répertoires de rep
du h rep

● Voir l'occupation du répertoire rep
du sh rep

Superviser le système, Charge système

●Le noyau gère l'ordonnancement de tous les processus du système

●Les processus peuvent être dans plusieurs états, les plus courants étant :
– R (Runnable) : le processus demande le processeur ou d'E/S
– S (Sleep) : le processus dort
– T (sTopped) : le processus a été arrêté (job control)

●La charge système («load average») est le nombre moyen de processus dans la run-queue (état 'R') pendant une période donnée

●La charge système ne reflète pas forcément la charge processeur

● La charge système ne tient pas compte du nombre de processeurs (ou de «cores»)

● uptime renvoie les load averages des 1, 5 et 15 dernières minutes :

user@host:~$ uptime
16:19:59 up 101 days, 6 users, load average: 5.10, 3.23, 2.31

(uptime donne aussi l'heure (!) et le temps de fonctionnement total du système)

● La charge est toujours disponible dans /proc/loadavg

user@host:~$ cat /proc/loadavg
5.10 3.23 2.31 4/40 20403

Superviser le système, Charge CPU

● vmstat permet de connaître la charge globale des processeurs

user@host:~$ vmstat

● mpstat permet de connaître la charge des processeurs :

user@host:~$ mpstat P ALL

lorsqu'un nombre n est ajouté en ligne de commande, l'affichage défile en continu toutes les n secondes

Superviser le système, Linux et la mémoire

● Le kernel utilise la mémoire pour :
– son propre code
– ses propres données
– le code des applications exécutées
– les données des applications exécutées
– le cache disque

● La mémoire en divisée en «pages»
● Les pages les moins utilisées sont mises en mémoire virtuelle (dans le swap), ce processus est appelé «swapping out» ou «paging out»
● Pour de bonnes performances, il faut, si possible, minimiser les opérations de page out

Superviser le système, Mémoire

● free permet de connaître l'état de la mémoire

user@host:~$ free m

● vmstat et free permettent aussi de connaître l'état de la mémoire virtuelle

user@host:~$ vmstat

● swapon s permet de connaître la liste des partitions utilisées pour le swap

user@host:~$ swapon s

Superviser le système, Couteaux suisses

● top : montre la liste des processus triée selon les critères choisis

● saidar : affiche un résumé très complet des différents sous-systèmes

● statgrab : permet d'obtenir des statistiques «formatées»

● sar permet de collecter des informations sur le système à intervalles réguliers ;
ces informations sont stockées sur le système et peuvent être analysées à posteriori

sar option s debut e fin

● Analyse « off-line » : sar

user@lazyserver:~$ sar u s 10:00:00 e 11:00:00

sar peut aussi monitorer en temps réel l'activité d'un processus : sar x PID 1 0

user@lazyserver:~$ sar x 6702 1 0

Superviser le système, Rebooter ?

● uptime donne la durée de fonctionnement du système

● shutdown h time arrête le système (equivalent à 'halt')

● shutdown r time reboote le système (equivalent à 'reboot')

● shutdown c annule un shutdown programmé
– time : hh:mm, +m ou now

Packages, Installation de logiciels

Packages, Installation de logiciels

Packages. Encore des chapelles !
● RPM sous Fedora, Redhat, Mandriva, SuSE...
● DEB sous Debian, Ubuntu
● tar.gz binaires sous Slackware
● scripts + tar.gz sources sous Gentoo
● OpenPKG, sorte d'espéranto du packaging
...sans compter les packages source

Packages, Redhat Package Manager

● Fonctionne sur les distributions basées RPM
http://en.wikipedia.org/wiki/List_of_Linux_distributions#RPM-based_free_distributions

● Deux commandes suffisent pour gérer les paquetages sur les systèmes basés RPM
– rpm : permet de manipuler les fichiers .rpm et de gérer les paquetages installés
– yum : permet d'installer des paquetages depuis différentes sources

Les packages RPM ont une extension de fichier <arch>.rpm (paquetages binaires) ou .src.rpm (paquetages source)

● La commande rpm accepte des URL (ftp/http)

● Les packages sont manipulés avec la seule commande rpm

● Installer un package
rpm i package.rpm

● Supprimer un package
rpm e package

● Mettre à jour un package installé
rpm u package.rpm

● Les options v et h rendent l'affichage plus 'humain'.

● La commande query (-q) permet d'interroger un package ou la base de données des packages installés

● Voir les informations sur un package
rpm qip package.rpm (query information)
rpm qi package (pour un package installé)

● Lister les fichiers d'un package
rpm qlp package.rpm (query list)
rpm ql package (pour un package installé)

● Rechercher dans quel package se trouve fichier
rpm qf fichier (query find)

Packages, yum

● La commande yum permet de rechercher ou d'installer un paquetage directement depuis un dépôt (Internet, CD/DVD, disque)

● La liste des dépôts est définie dans /etc/yum*

● Installer le paquetage ***nom***
yum install nom

● Rechercher les paquetages contenant la chaîne ***nom***
yum search nom

● Mettre à jour les informations depuis les dépôts
yum update

Packages, Packages Debian

● Fonctionne sur les distributions basées Debian
http://en.wikipedia.org/wiki/List_of_Linux_distributions#Debian-based_free_distributions

● Les systèmes debian possèdent l'équivalent (approximatif) du monde RPM pour gérer les paquetages :
– dpkg : permet de manipuler les fichiers .deb et de gérer les paquetages installés
– aptget : permet d'installer des paquetages depuis différentes sources

Les packages Debian ont une extension de fichier .deb (paquetages binaires)
La commande dpkg n'accepte pas d'URL

Packages, Packages Debian : dpkg

Les fichiers ***packages*** sont manipulés avec la commande dpkg

● Installer ou mettre à jour un package
dpkg i package.deb

● Supprimer un package
dpkg r package

● Supprimer un package et sa configuration
dpkg P package

● Reconfigurer un package
dpkgreconfigure package

● Rechercher un fichier dans les paquetages installés
dpkg S fichier

● Lister les fichiers actuellement installés pour un paquetage.
dpkg L paquetage

● Lister les fichiers installés par un paquetage.
dpkg l paquetage

● Donne l'état du paquetage installé
dpkg l paquetage

Packages, Packages Debian : aptget

● La commande aptget permet de rechercher ou d'installer un paquetage directement depuis un dépôt (Internet, CD/DVD, disque)

● aptget est très performant par rapport à yum

● La liste des dépôts est définie dans /etc/apt/{sources.list,sources.list.d/}

● Installer le paquetage ***nom*** aptget install ***nom***

Packages, Packages Debian : sources apt

● les outils de la famille apt utilisent des sources pour obtenir les packages et les métadonnées associées

● ces sources, définies dans /etc/apt/{sources.list,sources.list.d/} ont le format suivant :
deb http://fr.archive.ubuntu.com/ubuntu/ edgy main restricted

● Upgrader tous les paquetages installés
aptget upgrade

● Mettre à jour l'information depuis les dépots
aptget update

● Supprimer le paquetage
aptget remove package

● Passer à une distribution plus récente
aptget dist-upgrade

● Vérifier l'état de la base de données
aptget check

*Packages, Packages Debian : apt-**

● apt est en fait une famille de commandes dédiées à la gestion de paquetages.

● Quelques membres de la famille :
– aptfile : rechercher un fichier dans les dépots apt
– aptcache : rechercher un package dont la description ou le nom contient une expression régulière

aptfile search `which bash`
aptcache search tcpdump
aptcache n search '.*dump.*'

Packages, tar, {g,b}zip

● tar permet de créér un fichier contenant d'autres fichiers

● Il est souvent utilisé en combinaison avec gzip ou bzip2

● tar est parfois utilisé comme système de gestion de paquetage dans certaines distributions (Slackware)

● Créer une archive archive.tar contenant fichiers

tar cvf archive.tar fichiers

● Créér une archive tar compressée gzip ***archive.tar.gz*** contenant ***fichiers***

tar cvzf archive.tar.gz fichiers

● Créér une archive tar compressée bzip2 ***archive.tar.bz2*** contenant ***fichiers***

tar cvjf archive.tar.bz2 fichiers

● Lister le contenu de l'archive

tar tvf archive
tar tvzf archive.tar.gz
tar tvjf archive.tar.bz2

● Extraire une archive tar

tar xvf archive
tar xvzf archive.tar.gz
tar xvjf archive.tar.bz2

Boot process

Boot, les 5 étapes

(1) Boot hardware
exécution du bios
paramètres dans la NVRAM

(2) Chargeur d'OS
exécuté depuis le MBR par le BIOS
limité à 512 bytes

(3) Boot kernel
montage du root fs (/)
chargement des drivers
exécution d'init

(4) init
passage au runlevel demandé
exécution séquencée des scripts
console

(5) Scripts SysV
démarrage des services

(6) Boot hardware
exécution du bios
paramètres dans la NVRAM

(7) Chargeur d'OS
exécuté depuis le MBR par le BIOS
limité à 512 bytes

(8) Boot kernel
montage du root fs (/)
chargement des drivers
exécution d'init

(9) init
passage au runlevel demandé
exécution séquencée des scripts
console

(10) Scripts SysV
démarrage des services

Boot, Bootloaders : lilo, grub & co.

● Le travail du bootloader : charger le kernel en mémoire

● Il doit donc :
– savoir ou le trouver
– savoir le charger

● Après des années de domination de lilo, Grub à pris le dessus

● D'autres bootloaders existent (notamment syslinux, loadlin, silo, milo, emile, etc...)

● GRUB fonctionne en 3 temps :
– 1 er étage (512b d'assembleur) booté sur le MBR
– 1.5 ème étage (~10kb), booté au dela du MBR
– 2 ème étage (~100kb), chargé depuis /boot/grub

● GRUB sait donc (grâce à 1.5), lire un fichier sur un filesystem ext3/reiserfs/xfs/..

● Il lit aussi sa configuration directement sur le filesystem : pas besoin de «réinstaller» GRUB à chaque changement

● Le 2 ème étage trouve le kernel et lui passe la main

Boot, grub : menu.lst

● /boot/menu.lst est le fichier de configuration de grub

● il contient des scripts qui sont interprétés par grub lors du boot

● ces scripts peuvent être édités au boot si besoin
– démarrage en single user
– changement du root filesystem

● grub-install permet d'écrire les étages 1 et 1.5 sur le disque

sudo grubinstall /dev/sda

● la commande grub se présente comme un shell; par exemple les commandes équivalentes à grub-install sont :

[root@server ~]# grub

Boot, les 5 étapes

(1) Boot hardware
exécution du bios
paramètres dans la NVRAM

(2) Chargeur d'OS
exécuté depuis le MBR par le BIOS
limité à 512 bytes

(3) Boot kernel
montage du root fs (/)
chargement des drivers
exécution d'init

(4) init
passage au runlevel demandé
exécution séquencée des scripts
console

(5) Scripts SysV
démarrage des services

Boot, Kernel

● Le kernel se décompresse lui même en RAM, et s'exécute, passant par différentes phases :
– initialisation des sous-systèmes (mémoire, ACPI, CPU, ...)
– initialisation des drivers (IDE, SCSI, ...)
– chargement ramdisk
– chargement modules du ramdisk
– initialisation des (nouveaux) devices
– montage des pseudo-filesystems (/proc, /sys)
– montage de la (vraie) partition /
– exécution d'init

Kernel, Versions

Versionnement du noyau

Versions a.b.c[.d]
– a.b : branche principale
● b impair : branche de développement
2.1, 2.3, 2.5., ...
● b pair : branche stable
2.0, 2.2, 2.4, 2.6, …

– c : identifiant unique dans la série

– d : depuis 2.6.8/2.6.11, sous versions mineures intégrant des bugfixes (stabilisation de la version 'a.b.c')

uname r permet de connaître votre version du noyau

Kernel, Modules

● Les modules sont des « morceaux » de noyau

● Ils peuvent être chargés/déchargés pendant l'exécution du kernel

● Ils sont compilés à partir de sources du kernel… et peuvent aussi provenir de fournisseurs commerciaux (« drivers »)

● Un dictionnaire (modules.dep) permet de savoir quel module est nécessaire au fonctionnement de tel autre module

● On les trouve dans /lib/modules/`uname r`/

...interêt ?

● kernel plus petit (barre des 1,44Mo !)
● Empreinte mémoire optimisée
● Déboguage plus facile
● Modules binaires provenant de tiers
● Plusieurs version peuvent «cohabiter»
● Attention à la sécurité !

● Insertion d'un module dans le kernel

insmod /chemin/complet/vers/module
modprobe module

● Retrait d'un module du kernel

rmmod module

Runlevels

Runlevels, init

● init est le premier et le seul processus exécuté directement par le kernel (PID 1)
● il est le père de tous les autres
● du point de vue d'init, le système est dans des états discrets : les runlevels
● chaque runlevel définit un démarrage spécifique du système
● les runlevels possibles sont [0-6sS]
● le comportement d'init en fonction du runlevel est défini dans /etc/inittab
● init a donc pour vocation principale de mettre le système en état de marche

Runlevels, /etc/inittab

id:levels:action:process

● id : identifiant unique
● levels : les runlevels pour lesquels cette action s'applique
● action : action ou déclencheur
● process : le(s) processus à lancer

```
l0:0:wait:/etc/init.d/rc 0
l1:1:wait:/etc/init.d/rc 1
l2:2:wait:/etc/init.d/rc 2
l3:3:wait:/etc/init.d/rc 3
l4:4:wait:/etc/init.d/rc 4
l5:5:wait:/etc/init.d/rc 5
l6:6:wait:/etc/init.d/rc 6

# What to do when CTRLALTDEL is pressed.
ca:12345:ctrlaltdel:/sbin/shutdown t1 a r now
```

SysV init, Startup scripts

● init met le système en route en invoquant (indirectement) des scripts de démarrage
● Ces scripts permettent d'exécuter des applications persistantes (par exemple, un serveur de messagerie) appelées démons
● Ces scripts peuvent aussi exécuter des actions ponctuelles (montage des différents systèmes de fichiers, démarrage du réseau, ...)
● Ces scripts son situés dans des répertoires /etc/rcN.d (N : runlevel)
● Ces scripts sont en fait des liens symboliques vers de vrais scripts existant dans /etc/init.d

● Ils sont exécutés dans l'ordre de leur numéros de séquence

● Ce type d'initialisation est appelé « SysV style init » (N.d.T : initialisation à la sauce système cinq)

● Les scripts situés dans les différents répertoires correspondant aux runlevels suivent une convention de nommage précise :

[SK][09][09]label

● S/K : le script sera appelé avec le paramètre start ou stop (kill).
C'est la responsabilité du script d'accepter et argument et de le traiter convenablement.

● 00-99 : nombre à deux chiffres donnant le numéro de séquence.
Les scripts sont exécutés du plus petit numéro au plus grand.
L'unicité n'est pas requise.

● label : un label unique pour le script.
Correspond en général au nom entier du script original

SysV init, Runlevels

0 : arrêt de la machine

S/s/1 : single user
mode mono-utilisateur utilisé pour la maintenance
pas de réseau, personne ne se loggue

2 : mode multi-utilisateur variant selon la distribution
mode par défaut sous Ubuntu
mode multi-utilisateur sans réseau sous RedHat

3 : mode multi-utilisateur variant selon la distribution
mode texte par défaut sous Redhat/Fedora

4 : inutilisé sous la plupart des distributions

5 : mode multi-utilisateur
mode graphique par défaut sous Redhat/Fedora

6 : reboot de la machine

A l'exception des niveaux 0 et 6, ces conventions varient d'un unix à l'autre

La commande runlevel permet de connaître le runlevel précédent et
le runlevel actuel

SysV init, Startup scripts

$ ls /etc/rc2.d/

K77ntpserver
S01apport
S05vbesave
S10acpid
S10powernowd.early
S10sysklogd
S11klogd
S13gdm
S14ppp
S16openvpn
S18hplip
S19cupsys
S20apmd
S20dbus
S20festival
S20hotkeysetup
S20laptopmode
S20makedev
S20nbdclient
S20nbdserver
S20nvidiakernel
S20postfix
S20powernowd
S20rsync
S20ssh
S20sysstat
S20tftpdhpa
S20virtualbox
S20xinetd
S25bluetooth
S25mdadm
S89anacron
S89atd
S89cron
S90binfmtsupport
S95preload
S98usplash
S99acpisupport
S99rc.local
S89anacron
S89atd
S89cron
S90binfmtsupport
S95preload
S98usplash
S99acpisupport
S99rc.local
S99rmnologin

S99stopreadahead

SysV init, Démarrer un script 'à la main'

● Les applications qui doivent s'exécuter au boot installent un script dans /etc/init.d

● Il est ensuite possible de contrôler cette application grâce à ce script

● Démarrer l'application
/etc/init.d/application start

● Stopper l'application
/etc/init.d/application stop

● Redémarrer l'application
/etc/init.d/application restart

● Demander à l'application de recharger sa configuration
/etc/init.d/application reload

Runlevels, Changer de runlevel

● Au boot, il est possible de demander un démarrage mono-utilisateur à GRUB

● En ajoutant 'S' ou 'single' à la ligne kernel, le noyau va demander à init de démarrer en mode 'single user'

● En cours de fonctionnement, la commande telinit demande à init de modifier le runlevel
telinit *level*

shutdown h now (ou halt) est l'équivalent de telinit 0

shutdown r now (ou reboot) est l'équivalent de telinit 6

Cron

Cron, Fonctionnement

● cron permet de programmer des tâches récurrentes sur le système

● ces tâches sont listées dans des «crontabs»
– chaque utilisateur possède sa propre crontab
– il y a une crontab système

● ces fichiers sont scrutés par le démon cron/crond/anacron chaque minute (pas besoin de redémarrer le démon)

● cron exécute une tâche si son heure est venue

● grâce à cron, on peut automatiser la rotation des logs, la mise à jour de bases de données, les backups, la génération d'index…

Cron, Configuration

cron peut être configuré de multiples manières

● /etc/crontab
crontab système; contient les programmations globales du système

● /var/spool/cron/crontabs/$USER
contient la crontab de $USER

● /etc/cron.d/
contient des 'mini-crontabs' ajoutées par les packages à l'installation

● /etc/cron.{hourly,daily,weekly,monthly} contient des scripts exécutés respectivement toutes les heures/jours/semaines/mois

Cron, Format des crontabs

min heure jourmois mois joursemaine user command
(le champ user n'existe que dans la crontab système)

min : à quelle minute de l'heure [0-59]
heure : à quelle heure [0-23]
jourmois : quel jour du mois [1-31]
mois : quel mois de l'année [1-12]

joursemaine : quel jour de la semaine [0-7], 0=7=dimanche
user : sous quel utilisateur
command : commande à exécuter

Pour chaque champ chronologique, on peut avoir :
- une valeur, l'exécution aura lieu à cette valeur
- une plage de valeurs A-B, l'exécution aura lieu à chaque valeur
- une combinaison de plages séparée par des ',' (A-B,C,D-E,...)
- une '*', correspondant à la plage entière
- un modificateur '/n' pour une plage, provoquant l'exécution toutes les n fois

CHMOD

chmod 777 fichier -v ou bien : chmod u+w *fichier* -v

(avec « u » pris dans : « u » ou « g » ou « o » ou « ug » ou « uo » ou « go » ou « ug » ou « ugo » ; avec « w » pris dans « w » ou « r » ou « x » ; avec « + » pris dans « + » ou « - »

000	**- - -**	**- - -**	**- - -**
001	**- - -**	**- - -**	**- - x**
002	**- - -**	**- - -**	**- w -**
003	**- - -**	**- - -**	**- w x**
004	**- - -**	**- - -**	**r - -**
005	**- - -**	**- - -**	**r - x**
006	**- - -**	**- - -**	**r w -**
007	**- - -**	**- - -**	**r w x**
010	**- - -**	**- - x**	**- - -**
011	**- - -**	**- - x**	**- - x**
012	**- - -**	**- - x**	**- w -**
013	**- - -**	**- - x**	**- w x**
014	**- - -**	**- - x**	**r - -**
015	**- - -**	**- - x**	**r - x**
016	**- - -**	**- - x**	**r w -**
017	**- - -**	**- - x**	**r w x**
020	**- - -**	**- w -**	**- - -**
021	**- - -**	**- w -**	**- - x**
022	**- - -**	**- w -**	**- w -**
023	**- - -**	**- w -**	**- w x**
024	**- - -**	**- w -**	**r - -**
025	**- - -**	**- w -**	**r - x**
026	**- - -**	**- w -**	**r w -**
027	**- - -**	**- w -**	**r w x**
030	**- - -**	**- w x**	**- - -**

031 - - - - w x - - x
032 - - - - w x - w -
033 - - - - w x - w x
034 - - - - w x r - -
035 - - - - w x r - x
036 - - - - w x r w -
037 - - - - w x r w x

040 - - - r - - - - -
041 - - - r - - - - x
042 - - - r - - - w -
043 - - - r - - - w x
044 - - - r - - r - -
045 - - - r - - r - x
046 - - - r - - r w -
047 - - - r - - r w x

050 - - - r - x - - -
051 - - - r - x - - x
052 - - - r - x - w -
053 - - - r - x - w x
054 - - - r - x r - -
055 - - - r - x r - x
056 - - - r - x r w -
057 - - - r - x r w x

060 - - - r w - - - -
061 - - - r w - - - x
062 - - - r w - - w -
063 - - - r w - - w x
064 - - - r w - r - -
065 - - - r w - r - x
066 - - - r w - r w -
067 - - - r w - r w x

070 - - - r w x - - -

071	**- - -**	**r w x**	**- - x**
072	**- - -**	**r w x**	**- w -**
073	**- - -**	**r w x**	**- w x**
074	**- - -**	**r w x**	**r - -**
075	**- - -**	**r w x**	**r - x**
076	**- - -**	**r w x**	**r w -**
077	**- - -**	**r w x**	**r w x**

100	**- - x**	**- - -**	**- - -**
101	**- - x**	**- - -**	**- - x**
102	**- - x**	**- - -**	**- w -**
103	**- - x**	**- - -**	**- w x**
104	**- - x**	**- - -**	**r - -**
105	**- - x**	**- - -**	**r - x**
106	**- - x**	**- - -**	**r w -**
107	**- - x**	**- - -**	**r w x**
110	**- - x**	**- - x**	**- - -**
111	**- - x**	**- - x**	**- - x**
112	**- - x**	**- - x**	**- w -**
113	**- - x**	**- - x**	**- w x**
114	**- - x**	**- - x**	**r - -**
115	**- - x**	**- - x**	**r - x**
116	**- - x**	**- - x**	**r w -**
117	**- - x**	**- - x**	**r w x**
120	**- - x**	**- w -**	**- - -**
121	**- - x**	**- w -**	**- - x**
122	**- - x**	**- w -**	**- w -**
123	**- - x**	**- w -**	**- w x**
124	**- - x**	**- w -**	**r - -**
125	**- - x**	**- w -**	**r - x**
126	**- - x**	**- w -**	**r w -**
127	**- - x**	**- w -**	**r w x**
130	**- - x**	**- w x**	**- - -**
131	**- - x**	**- w x**	**- - x**
132	**- - x**	**- w x**	**- w -**
133	**- - x**	**- w x**	**- w x**
134	**- - x**	**- w x**	**r - -**
135	**- - x**	**- w x**	**r - x**
136	**- - x**	**- w x**	**r w -**
137	**- - x**	**- w x**	**r w x**

140	**- - x**	**r - -**	**- - -**
141	**- - x**	**r - -**	**- - x**
142	**- - x**	**r - -**	**- w -**
143	**- - x**	**r - -**	**- w x**
144	**- - x**	**r - -**	**r - -**
145	**- - x**	**r - -**	**r - x**
146	**- - x**	**r - -**	**r w -**
147	**- - x**	**r - -**	**r w x**
150	**- - x**	**r - x**	**- - -**
151	**- - x**	**r - x**	**- - x**
152	**- - x**	**r - x**	**- w -**
153	**- - x**	**r - x**	**- w x**
154	**- - x**	**r - x**	**r - -**
155	**- - x**	**r - x**	**r - x**
156	**- - x**	**r - x**	**r w -**
157	**- - x**	**r - x**	**r w x**
160	**- - x**	**r w -- - -**	
161	**- - x**	**r w -- - x**	
162	**- - x**	**r w -**	**- w -**
163	**- - x**	**r w -**	**- w x**
164	**- - x**	**r w -**	**r - -**
165	**- - x**	**r w -**	**r - x**
166	**- - x**	**r w -**	**r w -**
167	**- - x**	**r w -**	**r w x**
170	**- - x**	**r w x**	**- - -**
171	**- - x**	**r w x**	**- - x**
172	**- - x**	**r w x**	**- w -**
173	**- - x**	**r w x**	**- w x**
174	**- - x**	**r w x**	**r - -**
175	**- - x**	**r w x**	**r - x**
176	**- - x**	**r w x**	**r w -**
177	**- - x**	**r w x**	**r w x**

200	**- w -**	**- - -**	**- - -**
201	**- w -**	**- - -**	**- - x**
202	**- w -**	**- - -**	**- w -**
203	**- w -**	**- - -**	**- w x**
204	**- w -**	**- - -**	**r - -**
205	**- w -**	**- - -**	**r - x**
206	**- w -**	**- - -**	**r w -**
207	**- w -**	**- - -**	**r w x**
210	**- w -**	**- - x**	**- - -**
211	**- w -**	**- - x**	**- - x**
212	**- w -**	**- - x**	**- w -**
213	**- w -**	**- - x**	**- w x**
214	**- w -**	**- - x**	**r - -**
215	**- w -**	**- - x**	**r - x**
216	**- w -**	**- - x**	**r w -**
217	**- w -**	**- - x**	**r w x**
220	**- w -**	**- w -**	**- - -**
221	**- w -**	**- w -**	**- - x**
222	**- w -**	**- w -**	**- w -**
223	**- w -**	**- w -**	**- w x**
224	**- w -**	**- w -**	**r - -**
225	**- w -**	**- w -**	**r - x**
226	**- w -**	**- w -**	**r w -**
227	**- w -**	**- w -**	**r w x**
230	**- w -**	**- w x**	**- - -**
231	**- w -**	**- w x**	**- - x**
232	**- w -**	**- w x**	**- w -**
233	**- w -**	**- w x**	**- w x**
234	**- w -**	**- w x**	**r - -**
235	**- w -**	**- w x**	**r - x**
236	**- w -**	**- w x**	**r w -**
237	**- w -**	**- w x**	**r w x**

240 - w - r - - - - -
241 - w - r - - - - x
242 - w - r - - - w -
200 - w - r - - - w x
244 - w - r - - r - -
245 - w - r - - r - x
246 - w - r - - r w -
247 - w - r - - r w x

250 - w - r - x - - -
251 - w - r - x - - x
252 - w - r - x - w -
253 - w - r - x - w x
254 - w - r - x r - -
255 - w - r - x r - x
256 - w - r - x r w -
257 - w - r - x r w x

260 - w - r w -- - -
261 - w - r w -- - x
262 - w - r w - - w -
263 - w - r w - - w x
264 - w - r w - r - -
265 - w - r w - r - x
266 - w - r w - r w -
267 - w - r w - r w x

270 - w - r w x - - -
271 - w - r w x - - x
272 - w - r w x - w -
273 - w - r w x - w x
274 - w - r w x r - -
275 - w - r w x r - x
276 - w - r w x r w -
277 - w - r w x r w x

300	**- w x**	**- - -**	**- - -**
301	**- w x**	**- - -**	**- - x**
302	**- w x**	**- - -**	**- w -**
303	**- w x**	**- - -**	**- w x**
304	**- w x**	**- - -**	**r - -**
305	**- w x**	**- - -**	**r - x**
306	**- w x**	**- - -**	**r w -**
307	**- w x**	**- - -**	**r w x**
310	**- w x**	**- - x**	**- - -**
311	**- w x**	**- - x**	**- - x**
312	**- w x**	**- - x**	**- w -**
313	**- w x**	**- - x**	**- w x**
314	**- w x**	**- - x**	**r - -**
315	**- w x**	**- - x**	**r - x**
316	**- w x**	**- - x**	**r w -**
317	**- w x**	**- - x**	**r w x**
320	**- w x**	**- w -**	**- - -**
321	**- w x**	**- w -**	**- - x**
322	**- w x**	**- w -**	**- w -**
323	**- w x**	**- w -**	**- w x**
324	**- w x**	**- w -**	**r - -**
325	**- w x**	**- w -**	**r - x**
326	**- w x**	**- w -**	**r w -**
327	**- w x**	**- w -**	**r w x**
330	**- w x**	**- w x**	**- - -**
331	**- w x**	**- w x**	**- - x**
332	**- w x**	**- w x**	**- w -**
333	**- w x**	**- w x**	**- w x**
334	**- w x**	**- w x**	**r - -**
335	**- w x**	**- w x**	**r - x**
336	**- w x**	**- w x**	**r w -**
337	**- w x**	**- w x**	**r w x**

340	**- w x**	**r - -**	**- - -**
341	**- w x**	**r - -**	**- - x**
342	**- w x**	**r - -**	**- w -**
300	**- w x**	**r - -**	**- w x**
344	**- w x**	**r - -**	**r - -**
345	**- w x**	**r - -**	**r - x**
346	**- w x**	**r - -**	**r w -**
347	**- w x**	**r - -**	**r w x**
350	**- w x**	**r - x**	**- - -**
351	**- w x**	**r - x**	**- - x**
352	**- w x**	**r - x**	**- w -**
353	**- w x**	**r - x**	**- w x**
354	**- w x**	**r - x**	**r - -**
355	**- w x**	**r - x**	**r - x**
356	**- w x**	**r - x**	**r w -**
357	**- w x**	**r - x**	**r w x**
360	**- w x**	**r w -- - -**	
361	**- w x**	**r w -- - x**	
362	**- w x**	**r w -**	**- w -**
363	**- w x**	**r w -**	**- w x**
364	**- w x**	**r w -**	**r - -**
365	**- w x**	**r w -**	**r - x**
366	**- w x**	**r w -**	**r w -**
367	**- w x**	**r w -**	**r w x**
370	**- w x**	**r w x**	**- - -**
371	**- w x**	**r w x**	**- - x**
372	**- w x**	**r w x**	**- w -**
373	**- w x**	**r w x**	**- w x**
374	**- w x**	**r w x**	**r - -**
375	**- w x**	**r w x**	**r - x**
376	**- w x**	**r w x**	**r w -**
377	**- w x**	**r w x**	**r w x**

400	**r - -**	**- - -**	**- - -**
401	**r - -**	**- - -**	**- - x**
402	**r - -**	**- - -**	**- w -**
403	**r - -**	**- - -**	**- w x**
404	**r - -**	**- - -**	**r - -**
405	**r - -**	**- - -**	**r - x**
406	**r - -**	**- - -**	**r w -**
407	**r - -**	**- - -**	**r w x**
410	**r - -**	**- - x**	**- - -**
411	**r - -**	**- - x**	**- - x**
412	**r - -**	**- - x**	**- w -**
413	**r - -**	**- - x**	**- w x**
414	**r - -**	**- - x**	**r - -**
415	**r - -**	**- - x**	**r - x**
416	**r - -**	**- - x**	**r w -**
417	**r - -**	**- - x**	**r w x**
420	**r - -**	**- w -**	**- - -**
421	**r - -**	**- w -**	**- - x**
422	**r - -**	**- w -**	**- w -**
423	**r - -**	**- w -**	**- w x**
424	**r - -**	**- w -**	**r - -**
425	**r - -**	**- w -**	**r - x**
426	**r - -**	**- w -**	**r w -**
427	**r - -**	**- w -**	**r w x**
430	**r - -**	**- w x**	**- - -**
431	**r - -**	**- w x**	**- - x**
432	**r - -**	**- w x**	**- w -**
433	**r - -**	**- w x**	**- w x**
434	**r - -**	**- w x**	**r - -**
435	**r - -**	**- w x**	**r - x**
436	**r - -**	**- w x**	**r w -**
437	**r - -**	**- w x**	**r w x**

440	**r - -**	**r - -**	**- - -**
441	**r - -**	**r - -**	**- - x**
442	**r - -**	**r - -**	**- w -**
443	**r - -**	**r - -**	**- w x**
444	**r - -**	**r - -**	**r - -**
445	**r - -**	**r - -**	**r - x**
446	**r - -**	**r - -**	**r w -**
447	**r - -**	**r - -**	**r w x**
450	**r - -**	**r - x**	**- - -**
451	**r - -**	**r - x**	**- - x**
452	**r - -**	**r - x**	**- w -**
453	**r - -**	**r - x**	**- w x**
454	**r - -**	**r - x**	**r - -**
455	**r - -**	**r - x**	**r - x**
456	**r - -**	**r - x**	**r w -**
457	**r - -**	**r - x**	**r w x**
460	**r - -**	**r w -- - -**	
461	**r - -**	**r w -- - x**	
462	**r - -**	**r w -**	**- w -**
463	**r - -**	**r w -**	**- w x**
464	**r - -**	**r w -**	**r - -**
465	**r - -**	**r w -**	**r - x**
466	**r - -**	**r w -**	**r w -**
467	**r - -**	**r w -**	**r w x**
470	**r - -**	**r w x**	**- - -**
471	**r - -**	**r w x**	**- - x**
472	**r - -**	**r w x**	**- w -**
473	**r - -**	**r w x**	**- w x**
474	**r - -**	**r w x**	**r - -**
475	**r - -**	**r w x**	**r - x**
476	**r - -**	**r w x**	**r w -**
477	**r - -**	**r w x**	**r w x**

500	**r - x**	**- - -**	**- - -**
501	**r - x**	**- - -**	**- - x**
502	**r - x**	**- - -**	**- w -**
503	**r - x**	**- - -**	**- w x**
504	**r - x**	**- - -**	**r - -**
505	**r - x**	**- - -**	**r - x**
506	**r - x**	**- - -**	**r w -**
507	**r - x**	**- - -**	**r w x**
510	**r - x**	**- - x**	**- - -**
511	**r - x**	**- - x**	**- - x**
512	**r - x**	**- - x**	**- w -**
513	**r - x**	**- - x**	**- w x**
514	**r - x**	**- - x**	**r - -**
515	**r - x**	**- - x**	**r - x**
516	**r - x**	**- - x**	**r w -**
517	**r - x**	**- - x**	**r w x**
520	**r - x**	**- w -**	**- - -**
521	**r - x**	**- w -**	**- - x**
522	**r - x**	**- w -**	**- w -**
523	**r - x**	**- w -**	**- w x**
524	**r - x**	**- w -**	**r - -**
525	**r - x**	**- w -**	**r - x**
526	**r - x**	**- w -**	**r w -**
527	**r - x**	**- w -**	**r w x**
530	**r - x**	**- w x**	**- - -**
531	**r - x**	**- w x**	**- - x**
532	**r - x**	**- w x**	**- w -**
533	**r - x**	**- w x**	**- w x**
534	**r - x**	**- w x**	**r - -**
535	**r - x**	**- w x**	**r - x**
536	**r - x**	**- w x**	**r w -**
537	**r - x**	**- w x**	**r w x**

540	**r - x**	**r - -**	**- - -**
541	**r - x**	**r - -**	**- - x**
542	**r - x**	**r - -**	**- w -**
500	**r - x**	**r - -**	**- w x**
544	**r - x**	**r - -**	**r - -**
545	**r - x**	**r - -**	**r - x**
546	**r - x**	**r - -**	**r w -**
547	**r - x**	**r - -**	**r w x**
550	**r - x**	**r - x**	**- - -**
551	**r - x**	**r - x**	**- - x**
552	**r - x**	**r - x**	**- w -**
553	**r - x**	**r - x**	**- w x**
554	**r - x**	**r - x**	**r - -**
555	**r - x**	**r - x**	**r - x**
556	**r - x**	**r - x**	**r w -**
557	**r - x**	**r - x**	**r w x**
560	**r - x**	**r w -- - -**	
561	**r - x**	**r w -- - x**	
562	**r - x**	**r w -**	**- w -**
563	**r - -**	**r w -**	**- w x**
564	**r - x**	**r w -**	**r - -**
565	**r - x**	**r w -**	**r - x**
566	**r - x**	**r w -**	**r w -**
567	**r - x**	**r w -**	**r w x**
570	**r - x**	**r w x**	**- - -**
571	**r - x**	**r w x**	**- - x**
572	**r - x**	**r w x**	**- w -**
573	**r - x**	**r w x**	**- w x**
574	**r - x**	**r w x**	**r - -**
575	**r - x**	**r w x**	**r - x**
576	**r - x**	**r w x**	**r w -**
577	**r - x**	**r w x**	**r w x**

600	**r w -**	**- - -**	**- - -**
601	**r w -**	**- - -**	**- - x**
602	**r w -**	**- - -**	**- w -**
603	**r w -**	**- - -**	**- w x**
604	**r w -**	**- - -**	**r - -**
605	**r w -**	**- - -**	**r - x**
606	**r w -**	**- - -**	**r w -**
607	**r w -**	**- - -**	**r w x**
610	**r w -**	**- - x**	**- - -**
611	**r w -**	**- - x**	**- - x**
612	**r w -**	**- - x**	**- w -**
613	**r w -**	**- - x**	**- w x**
614	**r w -**	**- - x**	**r - -**
615	**r w -**	**- - x**	**r - x**
616	**r w -**	**- - x**	**r w -**
617	**r w -**	**- - x**	**r w x**
620	**r w -**	**- w -**	**- - -**
621	**r w -**	**- w -**	**- - x**
622	**r w -**	**- w -**	**- w -**
623	**r w -**	**- w -**	**- w x**
624	**r w -**	**- w -**	**r - -**
625	**r w -**	**- w -**	**r - x**
626	**r w -**	**- w -**	**r w -**
627	**r w -**	**- w -**	**r w x**
630	**r w -**	**- w x**	**- - -**
631	**r w -**	**- w x**	**- - x**
632	**r w -**	**- w x**	**- w -**
633	**r w -**	**- w x**	**- w x**
634	**r w -**	**- w x**	**r - -**
635	**r w -**	**- w x**	**r - x**
636	**r w -**	**- w x**	**r w -**
637	**r w -**	**- w x**	**r w x**

640	**r w -**	**r - -**	**- - -**
641	**r w -**	**r - -**	**- - x**
642	**r w -**	**r - -**	**- w -**
643	**r w -**	**r - -**	**- w x**
644	**r w -**	**r - -**	**r - -**
645	**r w -**	**r - -**	**r - x**
646	**r w -**	**r - -**	**r w -**
647	**r w -**	**r - -**	**r w x**
650	**r w -**	**r - x**	**- - -**
651	**r w -**	**r - x**	**- - x**
652	**r w -**	**r - x**	**- w -**
653	**r w -**	**r - x**	**- w x**
654	**r w -**	**r - x**	**r - -**
655	**r w -**	**r - x**	**r - x**
656	**r w -**	**r - x**	**r w -**
657	**r w -**	**r - x**	**r w x**
660	**r w -**	**r w -- - -**	
661	**r w -**	**r w -- - x**	
662	**r w -**	**r w -**	**- w -**
663	**r w -**	**r w -**	**- w x**
664	**r w -**	**r w -**	**r - -**
665	**r w -**	**r w -**	**r - x**
666	**r w -**	**r w -**	**r w -**
667	**r w -**	**r w -**	**r w x**
670	**r w -**	**r w x**	**- - -**
671	**r w -**	**r w x**	**- - x**
672	**r w -**	**r w x**	**- w -**
673	**r w -**	**r w x**	**- w x**
674	**r w -**	**r w x**	**r - -**
675	**r w -**	**r w x**	**r - x**
676	**r w -**	**r w x**	**r w -**
677	**r w -**	**r w x**	**r w x**

700	**r w x**	**- - -**	**- - -**
701	**r w x**	**- - -**	**- - x**
702	**r w x**	**- - -**	**- w -**
703	**r w x**	**- - -**	**- w x**
704	**r w x**	**- - -**	**r - -**
705	**r w x**	**- - -**	**r - x**
706	**r w x**	**- - -**	**r w -**
707	**r w x**	**- - -**	**r w x**
710	**r w x**	**- - x**	**- - -**
711	**r w x**	**- - x**	**- - x**
712	**r w x**	**- - x**	**- w -**
713	**r w x**	**- - x**	**- w x**
714	**r w x**	**- - x**	**r - -**
715	**r w x**	**- - x**	**r - x**
716	**r w x**	**- - x**	**r w -**
717	**r w x**	**- - x**	**r w x**
720	**r w x**	**- w -**	**- - -**
721	**r w x**	**- w -**	**- - x**
722	**r w x**	**- w -**	**- w -**
723	**r w x**	**- w -**	**- w x**
724	**r w x**	**- w -**	**r - -**
725	**r w x**	**- w -**	**r - x**
726	**r w x**	**- w -**	**r w -**
727	**r w x**	**- w -**	**r w x**
730	**r w x**	**- w x**	**- - -**
731	**r w x**	**- w x**	**- - x**
732	**r w x**	**- w x**	**- w -**
733	**r w x**	**- w x**	**- w x**
734	**r w x**	**- w x**	**r - -**
735	**r w x**	**- w x**	**r - x**
736	**r w x**	**- w x**	**r w -**
737	**r w x**	**- w x**	**r w x**

740	**r w x**	**r - -**	**- - -**
741	**r w x**	**r - -**	**- - x**
742	**r w x**	**r - -**	**- w -**
700	**r w x**	**r - -**	**- w x**
744	**r w x**	**r - -**	**r - -**
745	**r w x**	**r - -**	**r - x**
746	**r w x**	**r - -**	**r w -**
747	**r w x**	**r - -**	**r w x**
750	**r w x**	**r - x**	**- - -**
751	**r w x**	**r - x**	**- - x**
752	**r w x**	**r - x**	**- w -**
753	**r w x**	**r - x**	**- w x**
754	**r w x**	**r - x**	**r - -**
755	**r w x**	**r - x**	**r - x**
756	**r w x**	**r - x**	**r w -**
757	**r w x**	**r - x**	**r w x**
760	**r w x**	**r w -- - -**	
761	**r w x**	**r w -- - x**	
762	**r w x**	**r w -**	**- w -**
763	**r w x**	**r w -**	**- w x**
764	**r w x**	**r w -**	**r - -**
765	**r w x**	**r w -**	**r - x**
766	**r w x**	**r w -**	**r w -**
767	**r w x**	**r w -**	**r w x**
770	**r w x**	**r w x**	**- - -**
771	**r w x**	**r w x**	**- - x**
772	**r w x**	**r w x**	**- w -**
773	**r w x**	**r w x**	**- w x**
774	**r w x**	**r w x**	**r - -**
775	**r w x**	**r w x**	**r - x**
776	**r w x**	**r w x**	**r w -**
777	**r w x**	**r w x**	**r w**