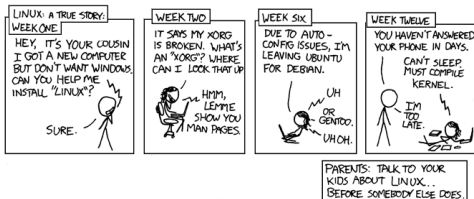


# Linux



# Sommaire

- 1 Installation
- 2 Paramétrage de l'environnement
- 3 Vi
- 4 Gérer le système de fichiers
- 5 Traitements des fichiers textes
- 6 Gestion de Processus
- 7 Planification de tâches
- 8 Utilisateurs
- 9 Permissions de fichiers
- 10 Système de fichiers
- 11 Arrêt et démarrage
- 12 Réseau TCP/IP sous GNU/Linux
- 13 Gestion des applications
- 14 Seb

# Installation

- 1 Installation
  - Introduction
  - Les grandes étapes
- 2 Paramétrage de l'environnement
- 3 Vi
- 4 Gérer le système de fichiers
- 5 Traitements des fichiers textes
- 6 Gestion de Processus
- 7 Planification de tâches
- 8 Utilisateurs
- 9 Permissions de fichiers
- 10 Système de fichiers
- 11 Arrêt et démarrage
- 12 Réseau TCP/IP sous GNU/Linux
- 13 Gestion des applications
- 14 Sch

# Principes

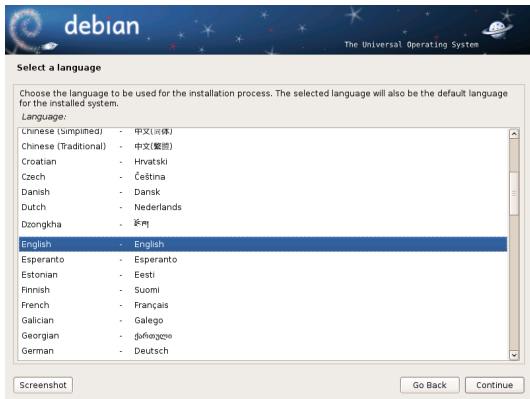
L'installation d'un système d'exploitation consiste à démarrer le système sur un périphérique de stockage externe, qui va charger un programme d'installation. Celui-ci va préconfigurer le système et installer les paquets logiciels choisis par l'intermédiaire d'un assistant.

Il est également possible d'automatiser le processus d'installation

# Les étapes de l'installation

- 1 Paramètres linguistiques.
- 2 Configuration du réseau.
- 3 Création de 2 comptes utilisateurs
- 4 Préparer les périphériques de stockage.
- 5 Installation des logiciels essentiels
- 6 Choix et installation des paquets.
- 7 Installation d'un chargeur de démarrage.
- 8 Redémarrer

# Paramètres linguistiques

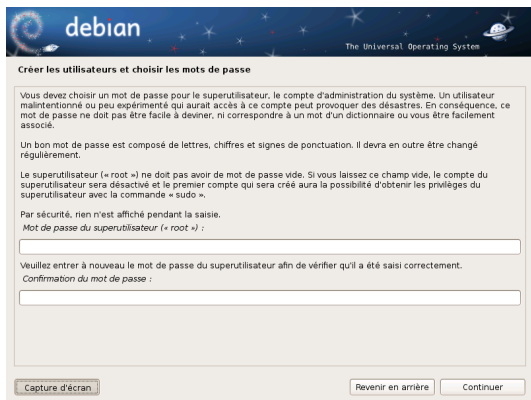


- ▶ Clavier
- ▶ Langue d'affichage
- ▶ Localisation géographique

# Réseau

- ▶ Choix du nom de machine et de domaine.

# Comptes



The screenshot shows the 'Créer les utilisateurs et choisir les mots de passe' (Create users and choose passwords) screen in the Debian installer. The header features the Debian logo and the text 'The Universal Operating System'. The main content area contains the following text:

**Créer les utilisateurs et choisir les mots de passe**

Vous devez choisir un mot de passe pour le superutilisateur, le compte d'administration du système. Un utilisateur malintentionné ou peu expérimenté qui aurait accès à ce compte peut provoquer des désastres. En conséquence, ce mot de passe ne doit pas être facile à deviner, ni correspondre à un mot d'un dictionnaire ou vous être facilement associé.

Un bon mot de passe est composé de lettres, chiffres et signes de ponctuation. Il devra en outre être changé régulièrement.

Le superutilisateur (« root ») ne doit pas avoir de mot de passe vide. Si vous laissez ce champ vide, le compte du superutilisateur sera désactivé et le premier compte qui sera créé aura la possibilité d'obtenir les privilèges du superutilisateur avec la commande « sudo ».

Par sécurité, rien n'est affiché pendant la saisie.

Mot de passe du superutilisateur (« root ») :

Veuillez entrer à nouveau le mot de passe du superutilisateur afin de vérifier qu'il a été saisi correctement.

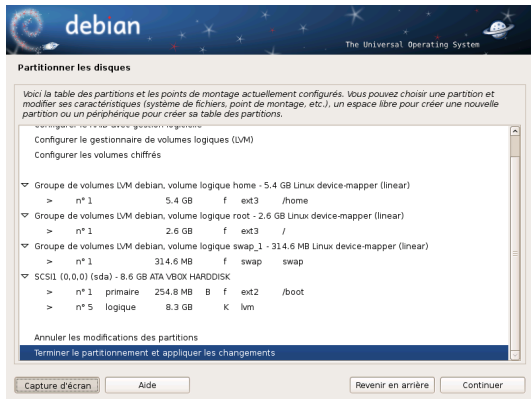
Confirmation du mot de passe :

At the bottom of the screen, there are three buttons: 'Capture d'écran', 'Revenir en arrière', and 'Continuer'.

- ▶ Compte root
- ▶ Compte utilisateur standard




# Partitionnement



- ▶ Partitionnement +/- assisté
- ▶ Possibilité de configurer LVM et RAID
- ▶ Possibilité de mettre en place un stockage chiffré

# Installation des paquets



- ▶ Installation des composants indispensables
- ▶ Choix par objectifs
- ▶ Choix des paquets individuels possible
- ▶ Installation minimale ! 

# Chargeur de démarrage



## ► Programme de démarrage

# Paramétrage de l'environnement

- 1 Installation
- 2 Paramétrage de l'environnement
  - Invite de commande
  - Raccourcis utiles
  - Configuration
  - Caractères spéciaux
  - Redirections
- 3 Vi
- 4 Gérer le système de fichiers
- 5 Traitements des fichiers textes
- 6
- 7 Planification de tâches
- 8 Utilisateurs
- 9 Permissions de fichiers
- 10 Système de fichiers
- 11 Arrêt et démarrage
- 12 Réseau TCP/IP sous GNU/Linux
- 13 Gestion des applications
- 14 Sch

# Introduction

Le shell est la principale interface entre vous et le système d'exploitation.

```
tom@workine:~$
```

- ▶ Il vous fournit un *environnement de travail*.
- ▶ Il interprète et exécute les commandes que vous lui indiquez, soit directement, soit par l'intermédiaire de scripts
- ▶ C'est le 1ier programme a être lancé après une connexion réussie.

# Invite de commande

L'invite de commande

```
tom@workine:~$
```

- ▶ Indique par sa présence que le système est en attente d'une entrée utilisateur.
- ▶ Fournie à l'utilisateur un certain nombre d'informations sur son environnement de travail

# Syntaxe d'une ligne de commande

Les commandes GNU/Linux obéissent *généralement* au schéma suivant:

```
tom@workine:~$ ls -al --color=auto /var/log
```

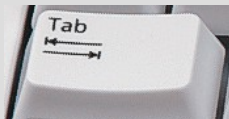
# Le shell vous facilite la vie

Quelle est 1<sup>ière</sup> touche la plus utilisée par un administrateur Linux?



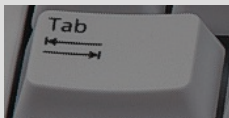
# Le shell vous facilite la vie

Quelle est 1<sup>ière</sup> touche la plus utilisée par un administrateur Linux?



# Le shell vous facilite la vie

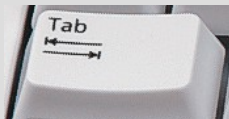
Quelle est 1<sup>ière</sup> touche la plus utilisée par un administrateur Linux?



```
$ oowr<TAB>
```

# Le shell vous facilite la vie

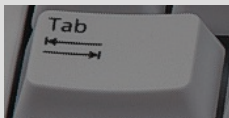
Quelle est 1<sup>ière</sup> touche la plus utilisée par un administrateur Linux?



```
$ oowriter
```

# Le shell vous facilite la vie

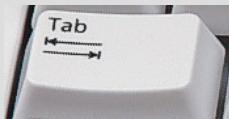
Quelle est 1<sup>ère</sup> touche la plus utilisée par un administrateur Linux?



```
$ oowriter /v<TAB>
```

# Le shell vous facilite la vie

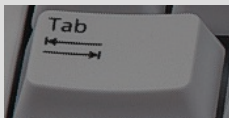
Quelle est 1<sup>ère</sup> touche la plus utilisée par un administrateur Linux?



```
$ oowriter /var/
```

# Le shell vous facilite la vie

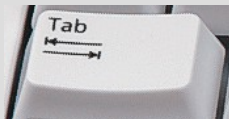
Quelle est 1<sup>ère</sup> touche la plus utilisée par un administrateur Linux?



```
$ oowriter /var/lo<TAB>  
local log lock
```

# Le shell vous facilite la vie

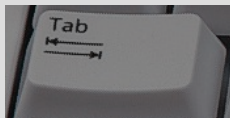
Quelle est 1<sup>ère</sup> touche la plus utilisée par un administrateur Linux?



```
$ oowriter /var/log/
```

# Le shell vous facilite la vie

Quelle est 1<sup>ère</sup> touche la plus utilisée par un administrateur Linux?

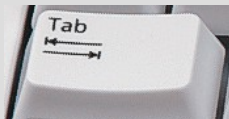


```
$ oowriter /var/log/mess<TAB>
```



# Le shell vous facilite la vie

Quelle est 1<sup>ère</sup> touche la plus utilisée par un administrateur Linux?



```
$ oowriter /var/log/messages
```

# Le shell vous facilite la vie: Historique

Le shell garde un historique des commandes.

Les touches **haut et bas** permettent de naviguer dans l'historique

# Le shell vous facilite la vie: Historique

Le shell garde un historique des commandes.

Les touches haut et bas permettent de naviguer dans l'historique

La séquence Ctrl+r+recherche permet de recherche dans l'historique

# Le shell vous facilite la vie: Historique

Le shell garde un historique des commandes.

Les touches **haut et bas** permettent de naviguer dans l'historique

La séquence **Ctrl+r+recherche** permet de recherche dans l'historique

La séquence **!!** rappelle la dernière commande exécutée.

# Le shell vous facilite la vie: Édition de la ligne de commande

En plus des touches fléchées, des raccourcis claviers permettent d'éditer facilement une ligne de commande

**Control+a** pour aller en début de ligne.

**Control+e** pour aller en fin de ligne.

**Control+w** pour effacer le mot *avant* le curseur.

**Control+k** efface tout à droite du curseur.

**Control+u** efface tout à gauche du curseur.

**Control+c** efface toute la ligne.

**Control+l** raccourci pour *clear*: efface l'écran.

# Le shell vous facilite la vie: Utilisation de la précédente commande

- ▶ `^erreur^correction` rappelle la précédente commande en remplaçant la chaîne *erreur* par *correction*.
- ▶ `!$` représente le dernier argument de la précédente commande
- ▶ `!*` représente l'ensemble des arguments de la précédente commande
- ▶ `!!: n` représente le nième argument de la précédente commande

# Le shell vous facilite la vie: les alias

```
tom@cafeine$ alias ll='/bin/ls -a --color=auto -c1'  
tom@cafeine$ ll  
. lesshist  
.  
. recently-used.xbel  
. ibam  
. gimp-2.4  
. viminfo  
. gtk-bookmarks  
. mysql_history
```

Les alias permettent de créer des raccourcis

- ▶ Pour les commandes complexes

## Le shell vous facilite la vie: les alias

```
tom@cafeine$ alias ll='/bin/ls -a --color=auto -c1'  
tom@cafeine$ ll  
.lessht  
.  
.recently-used.xbel  
.ibam  
.gimp-2.4  
.viminfo  
.gtk-bookmarks  
.mysql_history
```

Les alias permettent de créer des raccourcis

- ▶ Pour les commandes complexes
- ▶ et souvent utilisées.



# L'environnement

- ▶ Le shell représente votre environnement de travail
- ▶ Différentes méthodes permettent d'adapter cet environnement:
- ▶ Les alias
- ▶ Les *variables d'environnement*
  - ▶ La modification d'une VE entraîne un changement de comportement immédiat.
  - ▶ Elles sont en générales initialisées par l'intermédiaire des fichiers de configuration.
  - ▶ A moins d'être écrite dans le fichier de configuration adéquat, cette modification sera perdue à la prochaine fin de session.

# Les variables d'environnement

```
PS1 = \u @ \h : \w \s
```

```
tom@workine:~$
```

**PS1** permet de modifier l'apparence de l'invite de commande.

# Les variables d'environnement

**PATH** détermine les répertoires dans lesquels le shell va chercher les commandes que vous lui passez.

# Configuration du shell

- ▶ Lors de son lancement, le shell va lire un certain nombre de fichiers
- ▶ qui va lui permettre de mettre en place un environnement de travail adéquat.
  - ▶ Dans le cas d'un shell *de connexion*, il lira les fichiers `/etc/profile` et `~/.bash_profile`.
  - ▶ Dans le cas d'un shell non interactif, il lira les fichiers `~/.bashrc`

# Caractère spéciaux

Les caractères spéciaux sont *interprétés* par le shell.

| tube anonyme

|| "OU logique"

& exécution en arrière-plan

&& "ET logique"

; sépare plusieurs expressions

< et > caractères de redirection

## espace et tabulation

Si on souhaite utiliser ces caractères *textuellement*, on doit les *échapper* ou les *citer*.

# Jokers

```
tom@cafeine$ mv *.exe Windows/
```

Les jokers permettent de représenter un ou plusieurs caractères:

- \* représente n'importe quel chaîne de caractères

- ? représente n'importe quel caractère

- [a-d] n'importe quel caractère de l'intervalle spécifiée

- [abc] soit a, soit b, soit c

# Jokers

```
tom@cafeine$ mv *.mp? ~/Multimedia/
```

Les jokers permettent de représenter un ou plusieurs caractères:

- \* représente n'importe quel chaîne de caractères

- ? **représente n'importe quel caractère**

- [a-d] n'importe quel caractère de l'intervalle spécifiée

- [abc] soit a, soit b, soit c

# Jokers

```
tom@cafeine$ ls -cl [a-d]*
archivage.tex
communication.tex
beamerthemeopendoor.sty
```

Les jokers permettent de représenter un ou plusieurs caractères:

\* représente n'importe quel chaîne de caractères

? représente n'importe quel caractère

[a-d] n'importe quel caractère de l'intervalle spécifiée

[abc] soit a, soit b, soit c



# Jokers

```
tom@cafeine$ ls -c1 [ab]*
archivage.tex
beamerthemeopendoor.sty
tom@cafeine$
```

Les jokers permettent de représenter un ou plusieurs caractères:

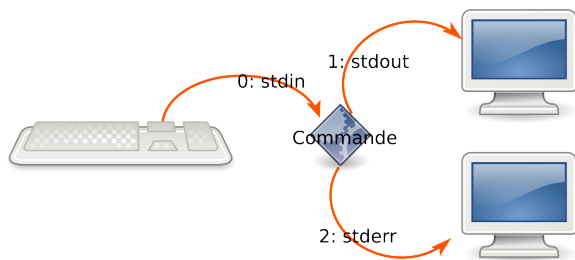
- \* représente n'importe quel chaîne de caractères

- ? représente n'importe quel caractère

- [a-d] n'importe quel caractère de l'intervalle spécifiée

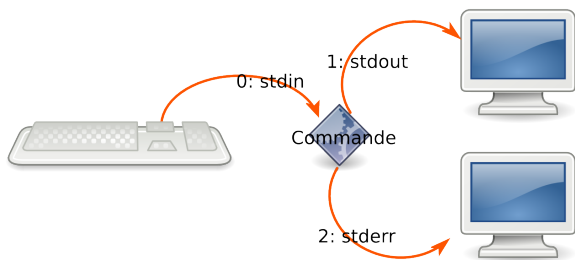
- [abc] soit a, soit b, soit c

# Entrées / Sorties et redirections



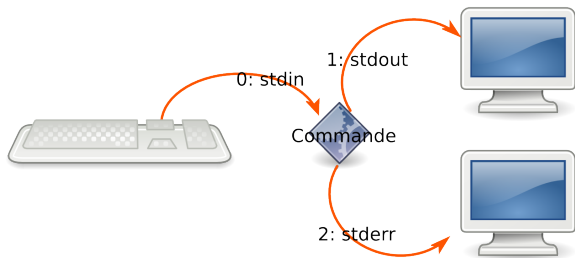
- ▶ La communication avec un programme se fait au moyen de *fichiers spéciaux*.

# Entrées / Sorties et redirections



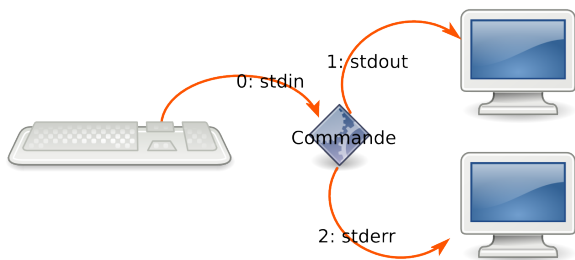
- ▶ La communication avec un programme se fait au moyen de *fichiers spéciaux*.
- ▶ Il est possible de *détourner* ces communications pour

# Entrées / Sorties et redirections



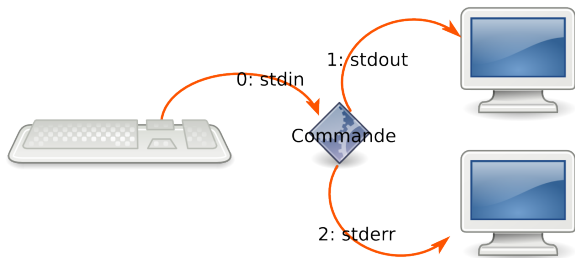
- ▶ La communication avec un programme se fait au moyen de *fichiers spéciaux*.
- ▶ Il est possible de *détourner* ces communications pour
  - ▶ récupérer le résultat d'une commande dans un fichier.

# Entrées / Sorties et redirections



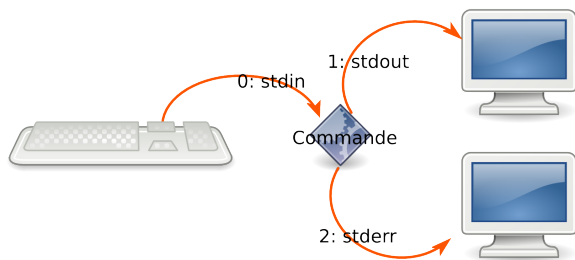
- ▶ La communication avec un programme se fait au moyen de *fichiers spéciaux*.
- ▶ Il est possible de *détourner* ces communications pour
  - ▶ récupérer le résultat d'une commande dans un fichier.
  - ▶ récupérer les erreurs issue de l'exécution d'une commande.

# Entrées / Sorties et redirections



- ▶ La communication avec un programme se fait au moyen de *fichiers spéciaux*.
- ▶ Il est possible de *détourner* ces communications pour
  - ▶ récupérer le résultat d'une commande dans un fichier.
  - ▶ récupérer les erreurs issue de l'exécution d'une commande.
  - ▶ renvoyer le résultat d'une commande à une autre commande.

# Entrées / Sorties et redirections



- ▶ La communication avec un programme se fait au moyen de *fichiers spéciaux*.
- ▶ Il est possible de *détourner* ces communications pour
  - ▶ récupérer le résultat d'une commande dans un fichier.
  - ▶ récupérer les erreurs issue de l'exécution d'une commande.
  - ▶ renvoyer le résultat d'une commande à une autre commande.
  - ▶ écrire des scripts non interactifs.

# Redirection dans un fichier

rediriger stdout: `ls -al > fichier_resultat`



# Redirection dans un fichier

rediriger stdout: `ls -al > fichier_resultat`

rediriger stderr: `ls -al 2> fichier_erreur`

# Redirection dans un fichier

rediriger stdout: `ls -al > fichier_resultat`

rediriger stderr: `ls -al 2> fichier_erreur`

rediriger les 2: `ls -al &> fichier`

# Redirection dans un fichier

rediriger stdout: `ls -al > fichier_resultat`

rediriger stderr: `ls -al 2> fichier_erreur`

rediriger les 2: `ls -al &> fichier`

ou: `ls -al > fichier 2>&1`

# Redirection dans un fichier

rediriger stdout: `ls -al > fichier_resultat`

rediriger stderr: `ls -al 2> fichier_erreur`

rediriger les 2: `ls -al &> fichier`

ou: `ls -al > fichier 2>&1`

## Redirection dans un fichier

rediriger stdout: `ls -al > fichier_resultat`

rediriger stderr: `ls -al 2> fichier_erreur`

rediriger les 2: `ls -al &> fichier`

ou: `ls -al > fichier 2>&1`

- ▶ Attention, si le fichier existe, son contenu sera *écrasé*

## Redirection dans un fichier

rediriger stdout: `ls -al > fichier_resultat`

rediriger stderr: `ls -al 2> fichier_erreur`

rediriger les 2: `ls -al &> fichier`

ou: `ls -al > fichier 2>&1`

- ▶ Attention, si le fichier existe, son contenu sera *écrasé*
- ▶ Sauf si on double le symbole de redirection `>>`

# Redirection dans un fichier

**rediriger stdout:** `ls -al > fichier_resultat`

**rediriger stderr:** `ls -al 2> fichier_erreur`

**rediriger les 2:** `ls -al &> fichier`

**ou:** `ls -al > fichier 2>&1`

- ▶ Attention, si le fichier existe, son contenu sera *écrasé*
- ▶ Sauf si on double le symbole de redirection `>>`
  - ▶ dans ce cas, les informations seront écrites à la fin du fichier.

# Redirection depuis un fichier

Elle consiste à remplacer l'entrée standard d'une commande par un fichier.

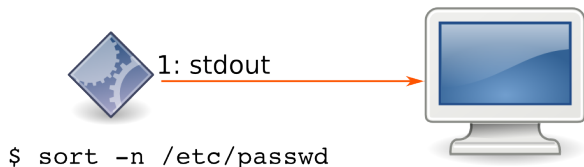
```
mysql < ~/base.sql
```



# Les tubes

- ▶ Les tubes sont un mécanisme permettant de passer le résultat d'une commande à une autre.
- ▶ Ils sont représentés par le symbole | (alt gr + 6)

# Les tubes



- ▶ Les tubes sont un mécanisme permettant de passer le résultat d'une commande à une autre.
- ▶ Ils sont représentés par le symbole | (alt gr + 6)

# Les tubes



- ▶ Les tubes sont un mécanisme permettant de passer le résultat d'une commande à une autre.
- ▶ Ils sont représentés par le symbole | (alt gr + 6)

# Redirections et tubes: la synthèse

- ▶ Si on souhaite enregistrer les résultats d'une commande dans un fichier

# Redirections et tubes: la synthèse

- ▶ Si on souhaite enregistrer les résultats d'une commande dans un fichier
- ▶ en conservant l'affichage sur la console.

# Redirections et tubes: la synthèse

- ▶ Si on souhaite enregistrer les résultats d'une commande dans un fichier
- ▶ en conservant l'affichage sur la console.
- ▶ on peut utiliser la commande *tee*:

# Redirections et tubes: la synthèse

- ▶ Si on souhaite enregistrer les résultats d'une commande dans un fichier
- ▶ en conservant l'affichage sur la console.
- ▶ on peut utiliser la commande *tee*:

```
$ ls -Alr > fichier
```

tout dans *fichier*, rien sur la console

# Redirections et tubes: la synthèse

- ▶ Si on souhaite enregistrer les résultats d'une commande dans un fichier
- ▶ en conservant l'affichage sur la console.
- ▶ on peut utiliser la commande *tee*:

```
$ ls -Alr | tee fichier
```

tout dans *fichier* **et** sur la console.



# Vi

- 1 Installation
- 2 Paramétrage de l'environnement
- 3 Vi**
- 4 Gérer le système de fichiers
- 5 Traitements des fichiers textes
- 6 Gestion de Processus
- 7 Planification de tâches
- 8 Utilisateurs
- 9 Permissions de fichiers
- 10 Système de fichiers
- 11 Arrêt et démarrage
- 12 Réseau TCP/IP sous GNU/Linux
- 13 Gestion des applications
- 14 Sch

# Introduction



- ▶ *vi* est un éditeur de texte
- ▶ disponible sur l'ensemble des systèmes de type unix / Linux.
- ▶ modulaire, il peut servir aussi bien comme éditeur de base, que comme plateforme complète de développement.
- ▶ Très puissant, sa maîtrise demande un peu d'habitude et de pratique.

## Question

Quelle est la 2ième touche la plus utilisée d'un clavier d'utilisateur linux?

## Question

Quelle est la 2ième touche la plus utilisée d'un clavier d'utilisateur linux?



# Particularités



```
echo hello world
test.sh          3,1          Bas
```

*vi* propose 2 modes:

- ▶ Un mode *commande* permettant de manipuler le texte dans son ensemble.
- ▶ Un mode *insertion* permettant de taper le texte.

Un 3ième mode *visuel* permet de faire des sélections.

# Particularités



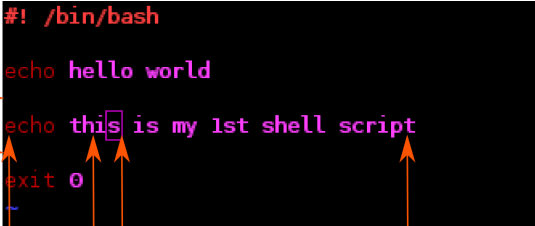
```
echo hello world
test.sh          3,1          Bas
-- INSERTION --
```

*vi* propose 2 modes:

- ▶ Un mode *commande* permettant de manipuler le texte dans son ensemble.
- ▶ Un mode *insertion* permettant de taper le texte.

Un 3ième mode *visuel* permet de faire des sélections.

## Pour passer d'un mode à l'autre



```
#!/bin/bash
echo hello world
echo this is my 1st shell script
exit 0
```

The screenshot shows a terminal window with a shell script. The text is as follows:

```
#!/bin/bash
echo hello world
echo this is my 1st shell script
exit 0
```

Annotations on the left side of the terminal:

- An orange arrow labeled 'O' points to the first line of the script.
- An orange arrow labeled 'o' points to the second line of the script.
- An orange arrow labeled 'i' points to the start of the second line.
- An orange arrow labeled 'a' points to the end of the second line.
- An orange arrow labeled 'A' points to the end of the second line.

Annotations at the bottom of the terminal:

- A vertical line labeled 'I' is at the start of the second line.
- A vertical line labeled 'i' is at the start of the second line.
- A vertical line labeled 'a' is at the end of the second line.
- A vertical line labeled 'A' is at the end of the second line.

on utilise

- ▶ les touches **i**, **I**, **o**, **O**, **a**, **A** pour passer du mode *commande* au mode insertion.

## Pour passer d'un mode à l'autre

```
#!/bin/bash
echo hello world
echo this is my 1st shell script
exit 0
```

on utilise

- ▶ les touches **i**, **I**, **o**, **O**, **a**, **A** pour passer du mode *commande* au mode insertion.
- ▶ la touche **echap** pour passer en mode *commande*.



## Pour passer d'un mode à l'autre

The screenshot shows a terminal window with a black background and red text. The script content is:
 

```
#!/bin/bash
echo hello world
echo this is my 1st shell script
exit 0
```

 Annotations include:
 

- Two orange arrows labeled 'O' pointing to the start of the first and second lines.
- Three orange arrows labeled 'i', 'I', and 'a' pointing to the start of the second line, with a small white box highlighting the word 'this'.
- One orange arrow labeled 'A' pointing to the end of the second line.

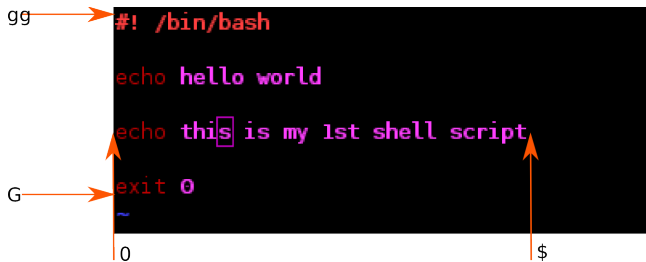
on utilise

- ▶ les touches **i**, **I**, **o**, **O**, **a**, **A** pour passer du mode *commande* au mode insertion.
- ▶ la touche **echap** pour passer en mode *commande*.
- ▶ Attention de ne pas mélanger les 2 modes!

# Édition et navigation

La touche **i** permet de passer en mode *insertion* et d'éditer le texte avec les touches habituelles. Les touches fléchées, Delete, backspace, Tab, PGUp, PGDown, etc. fonctionnent de manière habituelle.

# Navigation en mode commande



The image shows a terminal window with a black background and white text. The text is a shell script with the following lines: `#!/bin/bash`, `echo hello world`, `echo this is my 1st shell script`, and `exit 0`. There are four orange arrows pointing to specific parts of the script: one points to the shebang `#!/bin/bash`, one points to the `exit 0` command, one points to the space between `this` and `is` in the second `echo` line, and one points to the end of the second `echo` line. Below the terminal window, there are two orange arrows pointing to the characters `0` and `$` on the same horizontal level as the terminal's output.

```
gg → #! /bin/bash  
  
echo hello world  
echo this is my 1st shell script  
G → exit 0  
~  
0                                $
```

## Navigation en mode commande

```

gg → #! /bin/bash
     echo hello world
     echo this is my 1st shell script
G →  exit 0
     ~
     0                                     $
  
```

- ▶ Les éléments de *navigation* sont relatifs à la position du curseur.
- ▶ Ils peuvent s'appliquer à des *commandes* afin d'en spécifier *l'étendue*.
- ▶ Ils peuvent aussi être *quantifiés*.
- ▶ Ainsi, la séquence de touche <Esc>d3w va supprimer les 3 mots suivant le curseur. dG va supprimer toutes les lignes depuis le curseur jusqu'à la fin du fichier.

# Les commandes de suppression

**d**

# Les commandes de suppression

**d**elete - supprimer

# Les commandes de suppression

**d**delete - supprimer  
**c**

# Les commandes de suppression

**d**delete - supprimer  
**c**correct - corriger



## Suppression de texte

- x supprime le caractère sous le curseur
- dmouvement supprime *mouvement* (ex: d0 supprime tout du curseur jusqu'au début de la ligne).
- cmouvement est équivalent, mais passe en mode *insertion* après la suppression.

# Suppression de texte

- x supprime le caractère sous le curseur
- dmouvement supprime *mouvement* (ex: d0 supprime tout du curseur jusqu'au début de la ligne).
- cmouvement est équivalent, mais passe en mode *insertion* après la suppression.
- dd supprime l'ensemble de la ligne courante.
- cc supprime toute la ligne courante et passe ensuite en mode *insertion*.

## Suppression de texte

- x supprime le caractère sous le curseur
- dmouvement** supprime *mouvement* (ex: d0 supprime tout du curseur jusqu'au début de la ligne).
- cmouvement** est équivalent, mais passe en mode *insertion* après la suppression.
- dd** supprime l'ensemble de la ligne courante.
- cc** supprime toute la ligne courante et passe ensuite en mode *insertion*.
- D** supprime tout jusqu'à la fin de la ligne.
- C** idem, en passant ensuite en mode *insertion*

## Couper-Copier-coller

Sous vi, la suppression est en fait une “coupure”. On peut aussi copier de manière explicite:

**y**movement *copie* mouvement.

**Y** ou **yy** *copie* la ligne entière.

## Couper-Copier-coller

Sous vi, la suppression est en fait une “coupure”. On peut aussi copier de manière explicite:

**y**movement *copie* mouvement.

**Y** ou **yy** *copie* la ligne entière.

Et pour coller:

**p** permet de *coller* l'élément précédemment coupé à *droite* du curseur.

**P** *colle* l'élément à *gauche* du curseur.

# Annuler

vi propose un mécanisme d'annulation:

- u permet d'annuler la dernière opération
- . permet de répéter la dernière opération

**Control+r** revient dans l'historique des opérations (inverse de *u*).

# Quitter vi

Pour finir:

`:w` écrit les modifications apportées au fichier.

`:q` quitte vi

`:q!` quitte vi en abandonnant les modifications.

`:wq` combine les 2 1ieres commandes.

# Configuration

La configuration de *vim* est centralisée:

- ▶ dans `/etc/vim/vimrc` pour la configuration globale.
- ▶ dans `~/.vimrc` pour la configuration personnelle
- ▶ Le 2ième fichier étant interprété *après* le 1ier.



# Les principales directives du fichier personnel

```
filetype plugin on
syntax on
set showmode
set splitbelow
set splitright
set expandtab
set shiftwidth=3
set tabstop=3
set softtabstop=3
set smarttab
set smartindent
set foldmethod=indent
set nohlsearch
set showmatch
set binary noel
set backspace=indent,eol,start
set laststatus=2
set nocompatible
set visualbell
set ruler
set autochdir
set background=dark
set cursorline
set linebreak

set modeline
set modelines=5
"_F2_permet_de_passer_en_mode_(no)paste
" avec affichage dans la barre de statut
nnoremap <F2> :set invpaste paste?<CR>
inoremap <F2> <C-O><F2>
```

# Correction Orthographique

```

echo "Ceci est un tesst"

i=$RANDOM
( $i -lt 15000 ) && echo $i
exit 0

```

- ▶ Vim permet de détecter les erreurs d'orthographe en ignorant les mots-clé du langage utilisé.
- ▶ Il suffit de rajouter les instructions suivantes dans le fichier de configuration:

```

nnoemap <F3> :set invspell spell?<CR>
set spelllang=fr
augroup filetypedetect
  au BufNewFile,BufRead *.sh setlocal spell spelllang=fr
  au BufNewFile,BufRead *.c setlocal spell spelllang=fr
  au BufNewFile,BufRead *.tex setlocal spell spelllang=fr
augroup END

```

- ▶ L'activation / désactivation de la correction se fait par la touche <F3>.
- ▶ La commande z= affiche une liste de proposition de correction.

## Correction Orthographique

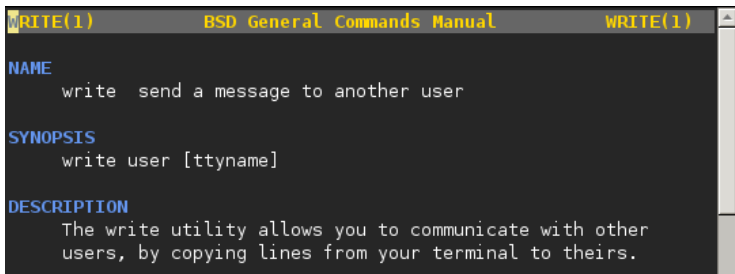
```
Remplacer "tesst" par :
1 "test"
2 "tests"
3 "t'est"
4 "est"
5 "lest"
```

- ▶ Vim permet de détecter les erreurs d'orthographe en ignorant les mots-clé du langage utilisé.
- ▶ Il suffit de rajouter les instructions suivantes dans le fichier de configuration:

```
nnoremap <F3> :set invspell spell?<CR>
set spelllang=fr
augroup filetypedetect
  au BufNewFile,BufRead *.sh setlocal spell spelllang=fr
  au BufNewFile,BufRead *.c setlocal spell spelllang=fr
  au BufNewFile,BufRead *.tex setlocal spell spelllang=fr
augroup END
```

- ▶ L'activation / désactivation de la correction se fait par la touche <F3>.
- ▶ La commande **z=** affiche une liste de proposition de correction.

## Page de man



```
WRITE(1) BSD General Commands Manual WRITE(1)

NAME
  write  send a message to another user

SYNOPSIS
  write user [ttyname]

DESCRIPTION
  The write utility allows you to communicate with other
  users, by copying lines from your terminal to theirs.
```

On peut configurer vim pour afficher la page de manuel d'un mot. Il suffit d'ajouter les instructions suivantes dans le fichier de configuration:

```
runtime ftplugin/man.vim
nnoremap K :Man <word><CR>
nnoremap k :Man 2 <word><CR>
```

- ▶ la touche "K" affiche la page de man du mot sous le curseur.
- ▶ la touche "k" affiche la page de man de l'appel-système sous le curseur.

## Page de man

```

WRITE(2)                               Linux Programmer's Manual                               WRITE(2)

NAME
    write - write to a file descriptor

SYNOPSIS
    #include <unistd.h>

    ssize_t write(int fd, const void *buf, size_t count);
  
```

On peut configurer vim pour afficher la page de manuel d'un mot. Il suffit d'ajouter les instructions suivantes dans le fichier de configuration:

```

runtime ftplugin /man.vim
nnoremap K :Man <word><CR>
nnoremap k :Man 2 <word><CR>
  
```

- ▶ la touche “K” affiche la page de man du mot sous le curseur.
- ▶ la touche “k” affiche la page de man de l'appel-système sous le curseur.

# Gérer le système de fichiers

- 1 Installation
- 2 Paramétrage de l'environnement
- 3 Vi
- 4 Gérer le système de fichiers**
  - Introduction
  - Répertoires
  - Attributs et types de fichiers
  - Commandes associées
  - Find
- 5 Traitements des fichiers textes
- 7 Planification de tâches
- 8 Utilisateurs
- 9 Permissions de fichiers
- 10 Système de fichiers
- 11 Arrêt et démarrage
- 12 Réseau TCP/IP sous GNU/Linux
- 13 Gestion des applications
- 14 Sch

# Gérer le système de fichiers



# Arborescence

- ▶ La structure d'un système de fichier Linux obéit au standard *FHS*.
- ▶ Les fichiers sont organisés en répertoires selon une structure *arborescente*.
- ▶ Le point de départ de cette arborescence est / (root, racine).



# Les répertoires

- ▶ Les répertoires sont les unités d'organisation de base d'un système Linux.
- ▶ Leur structure respecte une norme (le *FHS*):
  - /bin** contient les binaires essentiels au fonctionnement du système.
  - /boot** contient les fichiers nécessaires au démarrage du système.
  - /dev** fournit une représentation "fichier" des périphériques du système.
  - /etc** rassemble les fichiers de configuration du système et de ses composants et logiciels.
  - /home** centralise les répertoires personnels des utilisateurs du système.

# Les répertoires

`/usr` contient les programmes applicatifs installés sur le système (binaires, bibliothèques, documentation. . .)

`/sbin` `/et` `/usr/sbin` contiennent les commandes réservées à l'administration.

`/root` est le répertoire personnel de l'administrateur.

`/proc` et `/sys` sont une représentation "fichier" de l'état du système.

`/tmp` rassemble les données temporaires

`/var` contient les données des programmes

# Les répertoires

Dans chaque répertoire se trouvent 2 répertoires spéciaux:

- . fait référence au répertoire *courant*.
- .. fait référence au répertoire *parent*.

# Taille d'un répertoire

```
du [option] [fichier|repertoire]
```

```
du -h -c ~/Documents/Epsi
8,0K    /home/tom/Documents/Epsi/1 i / test
1,7M    /home/tom/Documents/Epsi/1 i
1,9M    /home/tom/Documents/Epsi/1A
3,6M    total
```

Affiche la taille du fichier ou du répertoire passé en paramètre. Principales options:

- h taille formatée pour une lecture facile.
- s n'affiche que la taille totale

# Taille du système de fichier

```
df [option] [systeme de fichier]
```

```
df -h
Sys. de fichiers  Taille  Uti. Disp.  Uti% Monte sur
/dev/sda          12G    5,2G  6,1G  46% /
...
turbine :/srv     368G   268G   82G  77% /srv
```

*df* affiche l'espace disponible / utilisé des différents systèmes de fichiers. L'option *-h* formate les tailles en Mo, Go, etc.

# Noms de fichiers

Quelques différences entre système Linux et système Microsoft:

- ▶ Les noms de fichiers sont *SenSibleS à lA cASsE*.
- ▶ L'extension n'est pas obligatoire et est d'ordre purement "cosmétique".
- ▶ Pour cacher un fichier, il faut préfixer son nom par un "."

# Lire les attributs d'un fichier

Un fichier unix est divisé en 2 parties:

- ▶ Les attributs ou *métadonnées*.
- ▶ Les données

Les principaux attributs sont visibles grâce à la commande `ls`

# La commande ls

```
tom@workine:~$ ls -l ~
drwx----- 2 tom maison 4,0K jui 23 08:33 PDF
drwxr-xr-x  2 tom maison 4,0K jan  9  2008 Templates
-rw-r--r--  1 tom maison 8,0K jui 23 09:47 console.png
drwxr-xr-x  2 tom maison 4,0K jui 18 11:29 public_html
...
```



# Les types de fichier

Le type de fichier est indiqué par le tout premier symbole du résultat de la commande ls.

```
tom@workine:~$ ls -l ~
drwx----- 2 tom maison 4,0K jui 23 08:33 PDF
drwxr-xr-x  2 tom maison 4,0K jan  9 2008 Templates
-rw-r--r--  1 tom maison 8,0K jui 23 09:47 console.png
drwxr-xr-x  2 tom maison 4,0K jui 18 11:29 public_html
...
```

Type:

- fichier régulier ou lien dur
- l lien symbolique
- d répertoire
- c périphérique en mode caractère
- b périphérique en mode bloc
- s socket
- p canal nommé

# Les attributs de date

- ▶ date
  - ▶ de modification (*mtime*).
  - ▶ d'accès (*atime*).
  - ▶ de modification des attributs (*ctime*).

## Copier, supprimer et renommer un fichier

**cp option source destination:** copie de fichier. L'option *-a* *préserve les attributs*.

**rm fichier:** supprime *fichier* **définitivement** après demande de confirmation.

**mv source destination:** déplace / renomme le fichier *source* en *destination*.

## Visualiser un fichier

`cat fichier` permet d'afficher l'intégralité du contenu de *fichier*.

`less fichier` permet:

- ▶ de naviguer dans le fichier à l'aide des touches fléchées ; gg ; G
- ▶ de faire une recherche: */terme recherché*.
- ▶ d'ouvrir le fichier dans un éditeur (touche v).

# La commande file

```
tom@workine:~$ file support.pdf
support.pdf: PDF document, version 1.4
```

**file option fichier:** permet d'afficher différentes informations relatives aux *données* du fichier

# La commande find

*find* permet de rechercher des fichiers en fonctions de leurs *attributs*.

`find chemin_depart expression`

**chemin\_depart:** dossier à partir duquel la recherche est faite.

**expression:** est composée d'*options*, d'*actions* et de *tests*

## Find: tests

Les *tests* permettent de limiter l'étendue d'une recherche aux fichiers répondant à des critères particuliers:

```
tom@cafeine$ find ~ -name "*.pdf" 
```

**-type d,f,l,...** recherche sur le type de fichier.

**-name nom** recherche sur le nom du fichier.

**-perm permissions** recherche sur les permissions.

**-size taille(bkM)** recherche sur la taille.

**-user nom:** recherche sur le propriétaire

## Find: éléments numériques pour les tests

lorsqu'un test prend une valeur *numérique*, celle-ci peut être spécifiée de la manière suivante:

```
tom@cafeine$ find . -cmin -60 -ls
```

**+num:** supérieur à *num*

**-num:** inférieur à *num*

**num:** égal à *num*



## Find: actions

Les *actions* s'appliquent à chaque fichier correspondant au résultat d'une recherche.

```
tom@cafeine$ find . -not -user tom -okdir mv  
{ } /tmp/ \;  
< mv ... ./fichier2 > ? y  
tom@cafeine$ █
```

**-print:** action par défaut. Affichage du nom de fichier.

**-ls:** Affichage détaillé du fichier et de ses attributs.

**-delete:** Suppression du fichier.

**-execdir command {} :** exécution de *commande* pour chaque fichier du résultat.

**-okdir command {}:** idem ci-dessus, en demandant confirmation.

## Find: erreurs courantes

**find: les chemins doivent précéder l'expression** : le *point de départ* de la recherche n'a pas été spécifié.

**find: paramètre manquant pour exec** : la commande `-exec` ne se termine pas par  
;

# Traitements des fichiers textes

- 1 Installation
- 2 Paramétrage de l'environnement
- 3 Vi
- 4 Gérer le système de fichiers
- 5 Traitements des fichiers textes**
  - Introduction
  - Commandes associées
  - Grep
  - Expressions régulières
  - Sed
- 7 Planification de tâches
- 8 Utilisateurs
- 9 Permissions de fichiers
- 10 Système de fichiers
- 11 Arrêt et démarrage
- 12 Réseau TCP/IP sous GNU/Linux
- 13 Gestion des applications
- 14 Sch

# Linux



# Introduction

L'administration sous GNU/Linux c'est:

- ▶ La manipulation de fichiers textes

# Introduction

L'administration sous GNU/Linux c'est:

- ▶ La manipulation de fichiers textes
- ▶ Au moyen de différents outils spécialisés

# Introduction

L'administration sous GNU/Linux c'est:

- ▶ La manipulation de fichiers textes
- ▶ Au moyen de différents outils spécialisés
- ▶ qu'il convient de combiner entre eux

# Introduction

L'administration sous GNU/Linux c'est:

- ▶ La manipulation de fichiers textes
- ▶ Au moyen de différents outils spécialisés
- ▶ qu'il convient de combiner entre eux
- ▶ pour obtenir le résultat escompté.



# Les filtres

Les commandes de filtre permettent de modifier l'affichage d'un fichier:

**more**, **less** et **pg** affichage *paginé* d'un fichier texte.

**head** et **tail** affiche le début ou la fin d'un texte.

**sort** tri

**cut** extraction de certaines parties d'un texte de longueur fixe

**grep** sélection de certaines lignes suivant un critère.

**sed** applique différents traitements au texte

# Les filtres

Les commandes de filtre permettent de modifier l'affichage d'un fichier:

**more**, **less** et **pg** affichage *paginé* d'un fichier texte.

**head** et **tail** affiche le début ou la fin d'un texte.

**sort** tri

**cut** extraction de certaines parties d'un texte de longueur fixe

**grep** sélection de certaines lignes suivant un critère.

**sed** applique différents traitements au texte

Ces commandes travaillent

- ▶ sur le fichier passé en paramètre (**less** /etc/passwd).
- ▶ sur les données fournies sur l'entrée standard (**cat** /etc/passwd | **less**)

# more et less

**more** permet d'afficher le contenu d'un fichier *page par page*

**less** propose les même fonctionnalités, avec en plus:

- ▶ La possibilité de revenir en arrière
- ▶ La possibilité de lancer des recherches (/recherche)
- ▶ La possibilité d'ouvrir le fichier pour édition

# tail et head

```
tom@workine:~$ tail -f /var/log/syslog
Jul 30 09:18:20 workine mountd[2726]: authenticated mount request from 192.168.
10.100:1016 for /home/Maison (/home)
Jul 30 09:18:20 workine mountd[2726]: authenticated mount request from 192.168.
10.100:768 for /home/Donnees (/home)
Jul 30 09:31:43 workine -- MARK --
Jul 30 09:51:43 workine -- MARK --
Jul 30 10:02:01 workine /USR/SBIN/CRON[21074]: (logcheck) CMD ( if [ -x /usr/
sbin/logcheck ]; then nice -n10 /usr/sbin/logcheck; fi)
```

**tail** affiche les 10 dernières lignes d'un fichier.

**tail -f** affiche ces 10 dernières lignes *en continu*.

**head** affiche les 10 premières lignes.

# tail et head

```
tom@workine:~$ tail -f /var/log/syslog
Jul 30 09:18:20 workine mountd[2726]: authenticated mount request from 192.168.
10.100:1016 for /home/Maison (/home)
Jul 30 09:18:20 workine mountd[2726]: authenticated mount request from 192.168.
10.100:768 for /home/Donnees (/home)
Jul 30 09:31:43 workine -- MARK --
Jul 30 09:51:43 workine -- MARK --
Jul 30 10:02:01 workine /USR/SBIN/CRON[21074]: (logcheck) CMD ( if [ -x /usr/
sbin/logcheck ]; then nice -n10 /usr/sbin/logcheck; fi)
```

**tail** affiche les 10 dernières lignes d'un fichier.

**tail -f** affiche ces 10 dernières lignes *en continu*.

**head** affiche les 10 premières lignes.

L'option **-n X**, commune aux 2 commandes, permet d'afficher X lignes.

# sort

```
tom@workine:~$ sort /etc/passwd
apt-mirror:x:110:121::/var/spool/apt-mirror:/bin/sh
avahi:x:108:116:Avahi mDNS daemon,,,:/var/run/avahi-daemon:/bin/false
backup:x:34:34:backup:/var/backups:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
games:x:5:60:games:/usr/games:/bin/sh
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/bin/sh
haldaemon:x:102:104:Hardware abstraction layer,,,:/var/run/hal:/bin/false
irc:x:39:39:ircd:/var/run/ircd:/bin/sh
```

*sort* permet de trier un flux de données.

Ses options sont:

- n tri numérique
- f ignore la casse de caractères
- r inverse l'ordre du tri

# cut

*cut* est utilisé pour extraire un ou plusieurs *champs*.

- ▶ un *champs* est un sous-ensemble d'une phrase divisée par un *séparateur*
- ▶ qui est le caractère "tabulation" par défaut.
- ▶ ce délimiteur peut être modifié par l'option "-d".
- ▶ le champ demandé est référencé par l'option "-f *position*" avec *position*:  
**nombre**: le champs dont la position est *nombre*.

# cut

*cut* est utilisé pour extraire un ou plusieurs *champs*.

- ▶ un champs est un sous-ensemble d'une phrase divisée par un *séparateur*
- ▶ qui est le caractère "tabulation" par défaut.
- ▶ ce délimiteur peut être modifié par l'option "-d".
- ▶ le champ demandé est référencé par l'option "-f *position*" avec position:  
**nombre**: le champs dont la position est *nombre*.  
**nombre1-nombre2** tous les champs dont la position est comprise dans l'intervalle nombre1 ~ nombre2



# cut

*cut* est utilisé pour extraire un ou plusieurs *champs*.

- ▶ un champs est un sous-ensemble d'une phrase divisée par un *séparateur*
- ▶ qui est le caractère "tabulation" par défaut.
- ▶ ce délimiteur peut être modifié par l'option "-d".
- ▶ le champ demandé est référencé par l'option "-f position" avec position:
  - nombre**: le champs dont la position est *nombre*.
  - nombre1-nombre2** tous les champs dont la position est comprise dans l'intervalle  $\text{nombre1} \sim \text{nombre2}$
  - n1,n2,n3**: tous les champs correspondant aux positions données

# cut: exemples

```
head -n1 /etc/passwd
```

```
root:0:0:root:/root:/bin/bash
```

# cut: exemples

```
cut -f3 -d: /etc/passwd
```

0

# cut: exemples

```
cut -f1,3 -d: /etc/passwd
```

```
root 0
```

# Recherches textuelles

## Question

Comment rechercher et remplacer de multiples occurrences d'une expression dans un ou plusieurs textes de manière efficace?

## Grep: recherche dans le contenu

*grep* permet de faire des recherches à l'intérieur des fichiers texte.

```
tom@workine:~$ grep tom /etc/passwd  
tom:x:1234:1234:~/home/tom:/bin/zsh
```

Ses principales options sont:

- r: recherche récursive
- l: affiche le nom du fichier contenant le terme recherché
- i: recherche insensible à la casse.
- v: inverse la condition
- o: n'affiche que la partie correspondant au critère
- c: affiche le nombre d'occurrence
- color affiche le résultat en couleur

# Les expressions régulières

## Question:

Comment exprimer le critère de recherche suivant de manière “formelle”?

*“Une phrase de caractères commençant par une MAJUSCULE suivie de 5 lettres, d’un tiret et de 3 chiffres”*

# Expressions régulières

Les *expressions régulières* permettent de représenter un *motif* de recherche de manière formelle.



# Expressions régulières

Les *expressions régulières* permettent de représenter un *motif* de recherche de manière formelle.

- ▶ Appelées aussi *expressions rationnelles*, les *regex* permettent d'affiner les critères de recherche.
- ▶ Utilisée notamment par *grep* et *sed*. Elles sont aussi disponibles via *man*, *vim*, *less* . . .
- ▶ Elles peuvent aussi servir de *conditions* dans les structures de tests.

# Caractères et classes

```
tom@cafeine$ grep --color 'Windows' grep_sed_exemple.txt
Objectif : A l'issue de ce cours, vous maîtriserez les fonctions élémentaires de Windows.
Vous serez capable de gérer vos fichiers à partir de Windows et lancer les applications bure
autiques présentes sur votre poste.
tom@cafeine$
```

- ▶ **a**: détecte le caractère donné
- ▶ **^** correspond au début d'une ligne
- ▶ **[abc]** correspond au caractère a, b ou c
- ▶ **[a-d]** n'importe quel caractère dans l'intervalle spécifiée.
- ▶ **.** correspond à n'importe quel caractère.
- ▶ **a\*** le caractère "a" 0 ou plusieurs fois
- ▶ **a+** le caractère "a" 1 ou plusieurs fois
- ▶ **a{n,m}** le caractère "a" répété n à m fois

# Caractères et classes

```
tom@cafeine$ sed -ne '/^Objectif/p' grep_sed_exemple.txt
Objectif : A l'issue de ce cours, vous maîtriserez les fonctions élémentaires de Windows.
tom@cafeine$
```

- ▶ a: détecte le caractère donné
- ▶ ^ correspond au début d'une ligne
- ▶ [abc] correspond au caractère a, b ou c
- ▶ [a-d] n'importe quel caractère dans l'intervalle spécifiée.
- ▶ . correspond à n'importe quel caractère.
- ▶ a\* le caractère "a" 0 ou plusieurs fois
- ▶ a+ le caractère "a" 1 ou plusieurs fois
- ▶ a{n,m} le caractère "a" répété n à m fois

# Caractères et classes

```
tom@cafeine$ grep --color "^[OP]" grep_sed_exemple.txt
Objectif : A l'issue de ce cours, vous maitriserez les fonctions élémentaires de Windows.
Public : Vous venez d'acquérir un ordinateur et vous souhaitez démarrer.
tom@cafeine$
```

- ▶ a: détecte le caractère donné
- ▶ ^ correspond au début d'une ligne
- ▶ [abc] correspond au caractère a, b ou c
- ▶ [a-d] n'importe quel caractère dans l'intervalle spécifiée.
- ▶ . correspond à n'importe quel caractère.
- ▶ a\* le caractère "a" 0 ou plusieurs fois
- ▶ a+ le caractère "a" 1 ou plusieurs fois
- ▶ a{n,m} le caractère "a" répété n à m fois

## Caractères et classes

```
tom@cafeine$ echo "toto likes linux" | grep --color "[k-o]"
toto likes linux
tom@cafeine$
```

- ▶ a: détecte le caractère donné
- ▶ ^ correspond au début d'une ligne
- ▶ [abc] correspond au caractère a, b ou c
- ▶ [a-d] n'importe quel caractère dans l'intervalle spécifiée.
- ▶ . correspond à n'importe quel caractère.
- ▶ a\* le caractère "a" 0 ou plusieurs fois
- ▶ a+ le caractère "a" 1 ou plusieurs fois
- ▶ a{n,m} le caractère "a" répété n à m fois

# Caractères et classes

```
tom@cafeine$ sed -e "/Win.* /d" grep_sed_exemple.txt
Objectif : A l'issue de ce cours, vous maîtriserez les fonctions élémentaires de Windows.

Public : Vous venez d'acquérir un ordinateur et vous souhaitez démarrer.

Niveau requis : Il serait préférable que vous ayez une première pratique du clavier et de la
souris.
tom@cafeine$ █
```

- ▶ a: détecte le caractère donné
- ▶ ^ correspond au début d'une ligne
- ▶ [abc] correspond au caractère a, b ou c
- ▶ [a-d] n'importe quel caractère dans l'intervalle spécifiée.
- ▶ . correspond à n'importe quel caractère.
- ▶ a\* le caractère "a" 0 ou plusieurs fois
- ▶ a+ le caractère "a" 1 ou plusieurs fois
- ▶ a{n,m} le caractère "a" répété n à m fois

# Caractères et classes

```
tom@cafeine$ echo "toto likes linux" | grep --color "li*"
toto likes linux
tom@cafeine$
```

- ▶ a: détecte le caractère donné
- ▶ ^ correspond au début d'une ligne
- ▶ [abc] correspond au caractère a, b ou c
- ▶ [a-d] n'importe quel caractère dans l'intervalle spécifiée.
- ▶ . correspond à n'importe quel caractère.
- ▶ a\* le caractère "a" 0 ou plusieurs fois
- ▶ a+ le caractère "a" 1 ou plusieurs fois
- ▶ a{n,m} le caractère "a" répété n à m fois

# Caractères et classes

```
(tom@workine)echo "to like linux" |grep --color "(to)+"  
to like linux
```

- ▶ a: détecte le caractère donné
- ▶ ^ correspond au début d'une ligne
- ▶ [abc] correspond au caractère a, b ou c
- ▶ [a-d] n'importe quel caractère dans l'intervalle spécifiée.
- ▶ . correspond à n'importe quel caractère.
- ▶ a\* le caractère "a" 0 ou plusieurs fois
- ▶ a+ le caractère "a" 1 ou plusieurs fois
- ▶ a{n,m} le caractère "a" répété n à m fois



# Caractères et classes

```
(tom@workine)echo "tototo like linux" |grep --color "(to){2}"  
tototo like linux
```

- ▶ a: détecte le caractère donné
- ▶ ^ correspond au début d'une ligne
- ▶ [abc] correspond au caractère a, b ou c
- ▶ [a-d] n'importe quel caractère dans l'intervalle spécifiée.
- ▶ . correspond à n'importe quel caractère.
- ▶ a\* le caractère "a" 0 ou plusieurs fois
- ▶ a+ le caractère "a" 1 ou plusieurs fois
- ▶ a{n,m} le caractère "a" répété n à m fois

## Autres:

- ▶ `ch(at|ien)` -> chat ou chien
- ▶ `[^a]` tout, sauf un a

# Classes de caractères POSIX

- ▶ `[: alnum:]`: tous les caractères alphanumériques, minuscules et majuscules.
- ▶ `[: blank:]`: tous les caractères d'espacement, tabulation, sauf les fins de ligne / paragraphes.
- ▶ `[: space:]`: tous les caractères de type "espace", y compris les fins de ligne
- ▶ `[: digit :]`: tous les chiffres.
- ▶ ...

# Capture

Il est possible d'extraire certaines parties d'une chaîne à l'aide des expressions régulières.

En effet, les parties entre parenthèses sont stockées dans des variables spéciales

- ▶ `<img src="([^\"]+)"`
- ▶ Cet exemple, appliqué à un fichier html, détecte toutes les images présentes sur la page.
- ▶ À l'aide des parenthèses, leur URL sera stockée dans la variable 1

## Shell et expressions régulières

Il est possible d'utiliser des regexp sans outil externe:

- ▶ \* correspond à n'importe quelle chaîne
- ▶ ? correspond à n'importe quel caractère
- ▶ [abc] soit a, b ou c
- ▶ [a - c] un caractère de l'intervalle a - c
- ▶ [^x] tout sauf x

Ex: lister tous les fichiers commençant par un chiffre, suivi de n'importe quelle lettre, sauf un m:

```
ls -l [0-9][^m]*
```

# Shell et expressions régulières

## Globbering étendu

```
shopt -s extglob
```

```
ls -l !(*.tex|*.pdf)
```

## Utilisation avec vi

Vi propose l'utilisation d'expressions régulières pour rechercher et modifier du texte.

```
:%s/^echo "(.*)"$ / printf  
  ( \1 ) ; /gc
```

```
[range]s/MOTIF/  
Remplacement/[options]
```

**range** : spécifie l'étendue de la recherche (% , . , \$)

**MOTIF** : l'expression régulière à rechercher.

**Remplacement** : la chaîne qui remplacera le motif recherché.

**Options** : options de recherche et remplacement, notamment *global* et *confirm*.

# Utilisation avec grep

```
grep -E "motif" fichier
```

**motif** : expression régulière représentant la chaîne de caractères à rechercher.

**fichier** : fichier dans lequel effectuer la recherche.

**Autres options** : `-color`, `-recursive`, `-ignore-case`, `-only-matching`



# Expressions régulières: exemples

Syntaxe: `grep --color -E regexp fichier(s)`

- ▶ ligne vide: `/^$/`

# Expressions régulières: exemples

Syntaxe: `grep --color -E regexp fichier(s)`

- ▶ ligne vide: `/^$/`
- ▶ toutes les lignes commençant par "un" et se terminant par "it": `^un.*it$`

# Expressions régulières: exemples

Syntaxe: `grep --color -E regexp fichier(s)`

- ▶ ligne vide: `/^$/`
- ▶ toutes les lignes commençant par "un" et se terminant par "it": `^un.*it$`
- ▶ tous les mots de 8 lettres ou plus: `\<[a-zA-Z]{8,}\>` ou `\<[:alpha:]{8,}\>`

# Expressions régulières: exemples

Syntaxe: `grep --color -E regexp fichier(s)`

- ▶ ligne vide: `/^$/`
- ▶ toutes les lignes commençant par "un" et se terminant par "it": `^un.*it$`
- ▶ tous les mots de 8 lettres ou plus: `\<[a-zA-Z]{8,}\>` ou `\<[:alpha:]{8,}\>`
- ▶ toutes les lignes commençant par "univ" et ne se terminant pas par "t" ou "y": `^univ.*[^ty]$`

## Sed: Modification du contenu

*sed* est un éditeur de flux.

- ▶ Il est principalement utilisé pour réaliser des “*recherche / replacements*” répétitifs
- ▶ Il travaille ligne par ligne
- ▶ Il propose 3 commandes principales:

## Sed: Modification du contenu

*sed* est un éditeur de flux.

- ▶ Il est principalement utilisé pour réaliser des “*recherche / remplacements*” répétitifs
- ▶ Il travaille ligne par ligne
- ▶ Il propose 3 commandes principales:
  - p: affiche
  - d: efface
  - s: remplace

## Sed: Modification du contenu

*sed* est un éditeur de flux.

- ▶ Il est principalement utilisé pour réaliser des “*recherche / remplacements*” répétitifs
- ▶ Il travaille ligne par ligne
- ▶ Il propose 3 commandes principales:
  - p: affiche
  - d: efface
  - s: remplace
- ▶ ces commandes peuvent s'appliquer à 1 ou plusieurs lignes du fichier:
  - n sur la neme ligne
  - n,m toutes les lignes entre la nieme et la mieme.
  - n,\$ depuis la neme ligne jusqu'à la dernière.
  - /regexp/ sur les lignes correspondant à l'expression.

## Sed: exemples

```
sed -ne '3p' exemple.txt
```

```
tom@workine:~$ sed -ne '3p' exemple.txt  
Vous serez capable de gérer vos fichiers à partir de Windows et lancer les appl  
ications bureautiques présentes sur votre poste.
```

L'option `-n` force sed à n'afficher que les lignes demandées.



# Sed: exemples

sed -e '1,3d' exemple.txt

```
tom@workine:~$ sed -e '1,3d' exemple.txt
```

```
Public : Vous venez d'acquérir un ordinateur et vous souhaitez démarrer.
```

```
Niveau requis : Il serait préférable que vous ayez une première pratique du clavier et de la souris.
```

## Sed: exemples

```
sed -ne '5,$p' exemple.txt
```

```
tom@workine:~$ sed -ne '5,$p' exemple.txt
```

```
Public : Vous venez d'acquérir un ordinateur et vous souhaitez démarrer.
```

```
Niveau requis : Il serait préférable que vous ayez une première pratique du clavier et de la souris.
```

Notez l'utilisation de l'option *-n*

## Sed: exemples

```
sed -e '/^ Objectif/d' exemple.txt
```

```
tom@workine:~$ sed -e '/^Objectif/d' exemple.txt
```

```
Vous serez capables de gérer vos fichiers à partir  
de windows et de lancer les applications bureauti  
ques présentes sur votre poste.
```

```
Public: vous venez d'acquérir un ordinateur et vou  
s souhaitez démarrer.
```

# Sed: Recherche et remplacement

À l'aide de la commande `s`, on peut spécifier un *critère de recherche*, suivi de la chaîne de substitution

# Sed: Recherche et remplacement

```
sed -e 's/[Ww]indows/Linux/' exemple.txt
```

```
tom@workine:~$ sed -e 's/[Ww]indows/Linux/' exemple.txt
```

```
Objectif : A l'issue de ce cours, vous maîtriserez les fonctions élémentaires de Linux.
```

```
Vous serez capable de gérer vos fichiers à partir de Linux et lancer les applications bureautiques présentes sur votre poste.
```

```
Public : Vous venez d'acquérir un ordinateur et vous souhaitez démarrer.
```

```
Niveau requis : Il serait préférable que vous ayez une première pratique du clavier et de la souris.
```

# Sed: Recherche et remplacement

Il est aussi possible de spécifier une étendue sur laquelle appliquer la substitution

```
sed -e '1s/[Ww]indows/Linux/'exemple.txt
```

```
tom@workine:~$ sed -e '1s/[Ww]indows/Linux/' exemple.txt
Objectif : A l'issue de ce cours, vous maîtriserez les fonctions élémentaires de Linux.

Vous serez capable de gérer vos fichiers à partir de Windows et lancer les applications bureautiques présentes sur votre poste.

Public : Vous venez d'acquérir un ordinateur et vous souhaitez démarrer.

Niveau requis : Il serait préférable que vous ayez une première pratique du clavier et de la souris.
```

# Gestion de Processus

- 1 Installation
- 2 Paramétrage de l'environnement
- 3 Vi
- 4 Gérer le système de fichiers
- 5 Traitements des fichiers textes
- 6 Gestion de Processus**
  - Introduction
  - Contrôle de tâches
  - Information sur les processus
  - Signaux
- 7 Planification de tâches
- 8 Utilisateurs
- 9 Permissions de fichiers
- 10 Système de fichiers
- 11 Arrêt et démarrage
- 12 Réseau TCP/IP sous GNU/Linux
- 13 Gestion des applications
- 14 Sch

# Définition

Un processus est une instance en fonctionnement d'un programme donné, caractérisé entre autre, par:

- ▶ Un numéro de processus unique: le pid
- ▶ Un processus père
- ▶ Un id utilisateur
- ▶ Un état



# Création de processus

- ▶ Principe de fonctionnement d'un shell lors du lancement d'une commande: fork + exec
- ▶ Il est possible de lancer une commande en *arrière-plan* en la suffixant par &.
- ▶ Attention: Vos processus auront au mieux la durée de vie de votre shell (donc de votre xterm).

# Création de processus

```
tom@workine:~$ a=0 ; while [[ $a -lt 1000 ]] ; do let "a+=1" ; done &  
[1] 7627  
tom@workine:~$ █
```

- ▶ Principe de fonctionnement d'un shell lors du lancement d'une commande: fork + exec
- ▶ Il est possible de lancer une commande en *arrière-plan* en la suffixant par &.
- ▶ Attention: Vos processus auront au mieux la durée de vie de votre shell (donc de votre xterm).

# Création de processus

```
tom@workine:~$ a=0 ; while [[ $a -lt 1000 ]] ; do let "a+=1" ; done &
[1] 7627
tom@workine:~$
[1]+  Done                    while [[ $a -lt 1000 ]] ; do
    let "a+=1";
done
```

- ▶ Principe de fonctionnement d'un shell lors du lancement d'une commande: fork + exec
- ▶ Il est possible de lancer une commande en *arrière-plan* en la suffixant par &.
- ▶ Attention: Vos processus auront au mieux la durée de vie de votre shell (donc de votre xterm).

# Fonctionnement

```
tom@workine:~$ vi test.sh
```

- ▶ Mise en arrière-plan d'un processus: Control+Z (passage à l'état suspendu ) et **bg** [%numero du job] ( **B**ack**G**round ).
- ▶ Lister les processus en arrière-plan: **jobs**.
- ▶ Remettre un processus en avant-plan: **fg** %numero du **jobs**(**F**ore**G**round).



# Fonctionnement

```
tom@workine:~$ vi test.sh
[2]+  Stopped                  vi test.sh
tom@workine:~$ █
```

- ▶ Mise en arrière-plan d'un processus: Control+Z (passage à l'état suspendu ) et **bg** [%numero du job] ( **B**ack**G**round ).
- ▶ Lister les processus en arrière-plan: **jobs**.
- ▶ Remettre un processus en avant-plan: **fg** %numero du **jobs**(**F**ore**G**round).

# Fonctionnement

```
tom@workine:~$ vi test.sh
[2]+  Stopped                  vi test.sh
tom@workine:~$ bash ./test.sh
hello world
tom@workine:~$ █
```

- ▶ Mise en arrière-plan d'un processus: Control+Z (passage à l'état suspendu ) et **bg** [%numero du job] ( **B**ack**G**round ).
- ▶ Lister les processus en arrière-plan: **jobs**.
- ▶ Remettre un processus en avant-plan: **fg** %numero du **jobs**(**F**ore**G**round).

# Fonctionnement

```
tom@workine:~$ vi test.sh
[2]+  Stopped                  vi test.sh
tom@workine:~$ bash ./test.sh
hello world
tom@workine:~$ fg
```

- ▶ Mise en arrière-plan d'un processus: Control+Z (passage à l'état suspendu ) et **bg** [%numero du job] ( **B**ack**G**round ).
- ▶ Lister les processus en arrière-plan: **jobs**.
- ▶ Remettre un processus en avant-plan: **fg** %numero du **jobs**(**F**ore**G**round).



# Fonctionnement

```
#!/bin/bash
echo hello world
~
~
~
~
~
~
~
~
~
~
```

- ▶ Mise en arrière-plan d'un processus: Control+Z (passage à l'état suspendu ) et **bg** [%numero du job] ( **B**ack**G**round ).
- ▶ Lister les processus en arrière-plan: **jobs**.
- ▶ Remettre un processus en avant-plan: **fg** %numero du **jobs**(**F**ore**G**round).

## Processus: affichage d'information

- ▶ La commande *ps* permet d'afficher la liste des processus en fonctionnement, ainsi qu'un certain nombre de caractéristiques. (*ps -ef* permet d'obtenir les infos suivantes: uid, pid, ppid, cpu, date de lancement du processus, console, temps cpu, commande)
- ▶ La commande *top* permet d'obtenir le même type d'information, en temps réel.
- ▶ La commande *vmstat* affiche des informations sur les processus, la mémoire vive, la mémoire virtuelle et les entrée/sorties.
- ▶ N'hésitez pas à consulter les pages de *man* correspondante pour plus d'info...

## ps -ef

Utilisateur à l'origine du processus

Identifiant du processus

Identifiant du processus père

Processeur occupé à l'exécution du processus

```
tom@workine:~$ ps -ef
UID          PID    PPID  C  STIME TTY          TIME CMD
root         1      0  0  Jul09 ?          00:00:09 init [2]
root         2      0  0  Jul09 ?          00:00:00 [kthreadd]
root         3      2  0  Jul09 ?          00:00:01 [migration/0]
root         4      2  0  Jul09 ?          00:00:14 [ksoftirqd/0]
root         5      2  0  Jul09 ?          00:00:03 [watchdog/0]
root         6      2  0  Jul09 ?          00:00:01 [migration/1]
root         7      2  0  Jul09 ?          00:00:17 [ksoftirqd/1]
root         8      2  0  Jul09 ?          00:00:00 [watchdog/1]
root         9      2  0  Jul09 ?          00:00:37 [events/0]
root        10      2  0  Jul09 ?          00:00:47 [events/1]
root        11      2  0  Jul09 ?          00:00:00 [khelper]
root        42      2  0  Jul09 ?          00:00:15 [kblockd/0]
```

Console à laquelle le processus est attaché

Temps CPU utilisé par le processus

Commande associée au processus

# Gestion des processus: signaux

Les signaux représentent le principal moyen de communication entre l'utilisateur et les processus.

- ▶ Pour envoyer un signal à un processus: **kill** -SIGNAL pid
- ▶ Vous ne pouvez pas tuer un processus qui n'est pas à vous ( sauf si vous êtes *root*).
- ▶ Le comportement par défaut d'un processus, à la réception d'un signal est de se terminer.

# Les signaux

Signaux: **kill** -l pour avoir la liste. Notamment:

- ▶ 2 - SIGINT - termine le processus, envoyé par ^C
- ▶ 9 - SIGKILL - force la terminaison du processus
- ▶ 15 - SIGTERM - termine le processus
- ▶ SIGUSR1 - signal défini par l'utilisateur
- ▶ SIGSTOP - mise en pause du processus ( passage en arrière-plan )
- ▶ SIGCONT - reprise de l'exécution

# Exercices d'application

- 1 À partir d'un terminal, lancez la commande "*vi exercice1.txt*"
- 2 Passez la commande en arrière-plan.
- 3 Récupérez, à l'aide de la commande "*ps -ef*", l'identifiant du processus que vous venez de créer.
- 4 Affichez la liste des processus tournant en tâches de fond.
- 5 Remettez votre processus *vi* en avant-plan.
- 6 Repassez le en arrière-plan.
- 7 Envoyez-lui le signal SIGTERM.
- 8 Notez les messages s'affichant sur la console.

# Planification de tâches

- 1 Installation
- 2 Paramétrage de l'environnement
- 3 Vi
- 4 Gérer le système de fichiers
- 5 Traitements des fichiers textes
- 6 Gestion de Processus
- 7 Planification de tâches**
  - Introduction
  - At
  - Cron
  - Anacron
- 8 Utilisateurs
- 9 Permissions de fichiers
- 10 Système de fichiers
- 11 Arrêt et démarrage
- 12 Réseau TCP/IP sous GNU/Linux

# Introduction

La **planification de tâches** , c'est l'exécution *régulières* de scripts et de commandes.

La **programmation** , c'est l'exécution d'un script ou d'une commande à une date et une heure *précise*.



# Introduction

La **planification de tâches** , c'est l'exécution *régulières* de scripts et de commandes.

La **programmation** , c'est l'exécution d'un script ou d'une commande à une date et une heure *précise*.

Ces fonctions, sont respectivement assurées par les services *cron* et *at*

# Résultat

Le résultat (c'est à dire son affichage) d'une tâche planifiée ou programmée est *envoyé par mail* au propriétaire de la tâche.

# Installation

Le paquet *at* permet d'installer et de démarrer le service.

*/etc/init.d/atd*: script d'init

*/var/spool/cron/atjobs*: répertoire de stockage des tâches.

*/var/spool/cron/atspools*: répertoire de stockage du résultat des tâches.

## Configuration du service

Le service accepte les options suivantes:

- l: charge cpu au delà de laquelle les tâches *at* ne se lanceront pas (1.5 par défaut).
- b: intervalle minimal entre 2 tâches. (60 sec par défaut)

Pour modifier ces valeurs, il faut modifier directement le *script d'init*.

# Sécurité

- ▶ Les tâches programmées via *at* s'exécutent avec les droits de l'utilisateur qui les programme.
- ▶ Il est possible de limiter l'utilisation du service via les fichiers:
  - /etc/at.allow*: si ce fichier existe, seuls les utilisateurs qui y sont listés peuvent utiliser le service.
  - /etc/at.deny*: si */etc/at.allow*, n'existe pas, on utilise ce fichier.

# Programmation

La programmation d'une tâche se fait par la commande `at`, qui lance un pseudo-éditeur nous permettant de rentrer la ou les commandes à programmer.

`at [TIME] [DATE]`

**TIME:** spécifie l'heure à laquelle la tâche sera exécutée. format *HHMM*. On peut aussi utiliser les mot-clés *midnight*, *noon*.

**DATE:** spécifie la date d'exécution. format *MMDDYY* ou *nom\_mois jour*.  
Mot-clés: *tomorrow*.

# Programmation

La programmation d'une tâche se fait par la commande `at`, qui lance un pseudo-éditeur nous permettant de rentrer la ou les commandes à programmer.

`at [TIME] [DATE]`

**TIME:** spécifie l'heure à laquelle la tâche sera exécutée. format *HHMM*. On peut aussi utiliser les mot-clés *midnight*, *noon*.

**DATE:** spécifie la date d'exécution. format *MMDDYY* ou *nom\_mois jour*.  
Mot-clés: *tomorrow*.

Options de la commande:

**-m** force l'envoi d'un mail, même s'il n'y a pas de résultat.

On quitte le pseudo-éditeur à l'aide de la séquence de touches *Crontrôle+d*.

# Affichage des tâches programmées

Seul *root* peut afficher les tâches de tous les utilisateurs.

`atq` affiche la liste des tâches.

`at -c °` affiche le détail d'une tâche.



# Suppression d'une tâche

Seul *root* peut supprimer une tâche dont il n'est pas propriétaire.

`atrm` ° supprime la tâche référencée par son °.

# Cron

Cron est le daemon chargé d'exécuter les tâches programmées à intervalle régulier.

- ▶ Il est normalement installé avec le système de base (paquet *cron*).
- ▶ Le script d'init est */etc/init.d/crond*
- ▶ Les informations d'exécution sont enregistrées dans les logs */var/log/auth.log* et */var/log/*
- ▶ Elles sont envoyées avec la *facility cron*, ce qui permet de les filtrer, au besoin.

# crontab

Cron se base sur le format de fichier *crontab* (man 5 crontab), composé de 6 champs:

- 1 minute (de 0 à 59)
- 2 heure (de 0 à 23)
- 3 jour du mois (de 1 à 31)
- 4 mois (de 1 à 12)
- 5 jour de la semaine (de 0 à 7, ou les noms)
- 6 commande à exécuter.

Les 5 premiers champs peuvent avoir pour valeur un nombre, une \*, une intervalle ou une fraction:

Ex: `00 */2 * * 1-5 /root/ scripts / stat .sh`

## Crontab utilisateur

Chaque utilisateur dispose de sa liste de tâches programmées, localisée dans `/var/spool/cron/crontabs/$USERNAME`.

Ce fichier est manipulé par la commande `crontab`, qui accepte les options suivantes:

- `crontab -l` affiche la liste des tâches planifiées.
- `crontab -e` édite la liste
- `crontab -r` supprime la liste
- `-u USER` permet d'effectuer la commande sur la liste des tâches de l'utilisateur `USER` (réservée à `root`).

# Crontab système

Le fichier `/etc/crontab` permet de planifier des tâches indépendamment d'un compte utilisateur.

- ▶ Ce fichier à la même structure qu'un crontab standard, avec un champs supplémentaire permettant de spécifier l'identifiant utilisateur sous lequel la commande va s'exécuter:
- ▶ Ce script se contente d'exécuter régulièrement les scripts présents dans différents répertoires.

## /etc/crontab

```
# /etc/crontab: system-wide crontab
# Unlike any other crontab you don't have
#   to run the 'crontab'
# command to install the new version when
#   you edit this file
# and files in /etc/cron.d. These files
#   also have username fields ,
# that none of the other crontabs do.

SHELL=/bin/sh
PATH=/usr/local/sbin:/usr/local/bin:/sbin
:/bin:/usr/sbin:/usr/bin

# m h dom mon dow user  command
17 * * * * root    cd / && run-parts
    --report /etc/cron.hourly
25 6 * * *    root    test -x /usr/sbin/
    anacron || ( cd / && run-parts --report
```

## Crontab logiciel

Certains paquets logiciels requierent l'installation d'une tâche planifiée pour assurer leur bon fonctionnement.

- ▶ Le système propose le répertoire */etc/cron.d* pour rassembler ces tâches planifiées.
- ▶ Chacune est rassemblée dans un fichier de ce répertoire, portant le nom du paquet.
- ▶ Ce fichier doit avoir la même structure que */etc/crontab*

Ex:

```
#  
# Regular cron jobs for the cron-apt  
# package  
#  
# Every night at 4 o'clock.  
0 4 * * * root test -x /usr/  
    sbin/cron-apt && /usr/sbin/cron-  
    apt  
# Every hour.
```

# Anacron

Le paquet *Anacron* permet de programmer des tâches de manière irrégulière.

- ▶ Lors de son exécution, il vérifie les tâches listées dans le fichier */etc/anacrontab*
- ▶ À chaque tâche est associée une période  $p$  et un délai  $d$ .
- ▶ Si la tâche n'a pas été exécutée depuis  $p$  jours, elle est lancée.
- ▶  $d$  correspond au délai à attendre avant de lancer la tâche.
- ▶ Lorsqu'une tâche est exécutée, *anacron* enregistre la date d'exécution.



## /etc/anacrontab

```
# /etc/anacrontab: configuration file for
    anacron

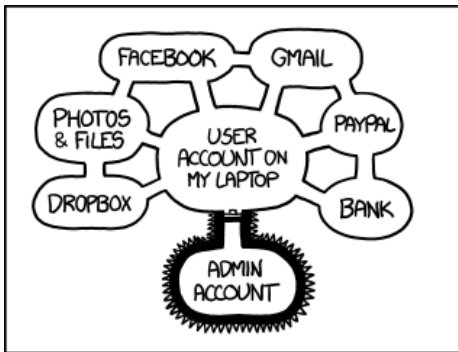
# See anacron(8) and anacrontab(5) for
    details.

SHELL=/bin/sh
PATH=/usr/local/sbin:/usr/local/bin:/sbin
    :/bin:/usr/sbin:/usr/bin

# These replace cron's entries
1      5      cron.daily      nice run-
    parts --report /etc/cron.daily
7      10     cron.weekly     nice run-
    parts --report /etc/cron.weekly
@monthly 15     cron.monthly nice
    run-parts --report /etc/cron.monthly
```

# Utilisateurs

- 1 Installation
- 2 Paramétrage de l'environnement
- 3 Vi
- 4 Gérer le système de fichiers
- 5 Traitements des fichiers textes
- 6 Gestion de Processus
- 7 Planification de tâches
- 8 Utilisateurs**
  - Introduction
  - Groupes
  - Utilisateurs
  - Mots de passe
- 9 Permissions de fichiers
- 10 Système de fichiers
- 11 Arrêt et démarrage
- 12 Réseau TCP/IP sous GNU/Linux



IF SOMEONE STEALS MY LAPTOP WHILE I'M LOGGED IN, THEY CAN READ MY EMAIL, TAKE MY MONEY, AND IMPERSONATE ME TO MY FRIENDS,  
BUT AT LEAST THEY CAN'T INSTALL DRIVERS WITHOUT MY PERMISSION.

# Introduction

- ▶ GNU/Linux propose un mécanisme sophistiqué de gestion des permissions sur les fichiers et répertoires.
- ▶ Chaque utilisateur du système se voit attribuer un identifiant unique (*uid*).
- ▶ Les programmes partagent l'*uid* de l'utilisateur qui les a lancés.
- ▶ Les droits d'accès sont déterminés en fonction de cet uid et des groupes auxquels l'utilisateur appartient.

# Groupes

- ▶ Les groupes permettent de rassembler des comptes utilisateurs de manière logique, afin d'en simplifier l'administration.
- ▶ De la même manière que les comptes utilisateurs, ils sont représentés par un entier unique.
- ▶ Les informations sont stockées dans le fichier `/etc/group`:
  - 1 nom du groupe
  - 2 mot de passe ( obsolète )
  - 3 GID identifiant numérique
  - 4 utilisateur1,utilisateur2,utilisateur3,...

# Commande de gestion des groupes

- 1 ajout d'un groupe: `groupadd`

# Commande de gestion des groupes

- 1 ajout d'un groupe: `groupadd`
- 2 suppression: `groupdel group`

# Commande de gestion des groupes

- 1 ajout d'un groupe: `groupadd`
- 2 suppression: `groupdel groupe`
- 3 ajout d'un utilisateur à un groupe: `gpasswd -a login groupe`



# Utilisateurs

- ▶ Unix est un système multi-utilisateurs.
- ▶ Chaque utilisateur est identifié par un numéro.
- ▶ root correspond à l'utilisateur administrateur. Son id est 0.
- ▶ Les comptes dont l'id est inférieur à 500 sont des comptes systèmes.
- ▶ A chaque compte utilisateur est associé un login, un répertoire personnel et un shell
- ▶ Ces informations sont renseignées dans le fichier /etc/passwd

# Types de compte

Les types de comptes Unix sont les suivant:

**root** : compte administrateur, dont *l'uid* est 0.

**comptes systèmes**: comptes utilisateur associés à des *services*. Il n'est en général pas possible de se connecter à la machine en utilisant ces comptes, car aucun mot de passe ne leur est attribué.

**compte utilisateur normal**

**compte désactivé** : Le compte ne peut *plus* être utilisé pour se connecter.

## Le fichier `/etc/passwd`

Fichier contenant les informations sur les comptes utilisateurs.

```
tom@cafeine$ cat /etc/passwd
tom:x:1234:1234:Thomas Constans,,,:/home/tom
:/bin/zsh
```

- ▶ 7 champs séparés par “ : ”
  - 1 login
  - 2 mot de passe chiffré ( optionnel )
  - 3 id numérique
  - 4 id numérique de group
  - 5 Nom complet / commentaire
  - 6 répertoire personnel
  - 7 shell
- ▶ Doit être lisible par tout le monde -> le mot de passe n'est plus stocké dans ce fichier.

# Création d'un compte utilisateur

Elle se fait à l'aide de la commande *useradd*: `useradd option login` avec comme option possible:

- d **repertoire** : Utiliser *repertoire* comme répertoire personnel de l'utilisateur.
  - m pour créer ce répertoire
- g **groupe\_principal** : définition du groupe principal de l'utilisateur
- s **/bin/bash** shell de l'utilisateur
- ...

## Création d'un utilisateur

La plupart des options de la commande *useradd* peut être fixée dans le fichier */etc/default/useradd*

```
GROUP=100  
HOME=/home  
INACTIVE=-1  
EXPIRE=  
SHELL=/bin/bash  
SKEL=/etc/skel  
CREATE_MAIL_SPOOL=yes
```

# Suppression d'un compte

À l'aide de `userdel` option **login**

# Suppression d'un compte

À l'aide de `userdel` option **login**

`-r` supprime le répertoire utilisateur

# Informations d'authentification

Ces informations sont désormais déportées dans le fichier `/etc/shadow`:

```
gdm:*:14130:0:99999:7:::  
openldap:!:14138:0:99999:7:::  
toto:$1$5EpzrHQD$EqglMzfSkQDqtgLW1gHI80:14186:0:99999:7:::
```

- 1 login.
- 2 mot de passe chiffré.
- 3 nombre de jours depuis le dernier changement de mot de passe, à compter du 01/01/1970.
- 4 délai avant de pouvoir changer le mot de passe.
- 5 durée (en jours) de validité du mot de passe.
- 6 durée (en jours) avant la date de fin de validité, pendant lesquels l'utilisateur est averti.
- 7 nombre de jours, une fois passée la date de validité, au-delà duquel le compte sera désactivé.
- 8 nombre de jours, à compter du 01/01/1970, depuis que le compte est désactivé.



# Informations d'authentification

Les données du fichier `/etc/shadow` sont modifiables par la commande `passwd`.

`passwd` permet de changer votre mot de passe.

`passwd utilisateur` permet de changer le mot de passe d'*utilisateur*.

`passwd --lock login` verrouille le compte de *login*.

`passwd --maxdays jours login` définit la durée de validité du mot de passe.

`passwd --status login` affiche l'état des données d'authentification de *login*.

## Informations d'authentification

La commande `passwd` et la valeur par défaut de ses options peuvent être réglées via le fichier `/etc/login.defs`

MAIL_DIR	/var/spool/mail
PASS_MAX_DAYS	99999
PASS_MIN_DAYS	0
PASS_MIN_LEN	5
PASS_WARN_AGE	7
UID_MIN	1000
UID_MAX	60000
SYS_UID_MIN	201
SYS_UID_MAX	999
GID_MIN	1000
GID_MAX	60000

# Utilisateurs: commandes associées

```
tom@cafeine:~$ id
uid=1234(tom) gid=1234(maison) groupes=4(adm),20(dialout),24(cdrom),29(audio),40(src),44(video),46(plugdev),105(netdev),107(fuse),109(scanner),1234(maison)
tom@cafeine:~$
```

**id** affiche votre identifiant numérique et les groupes auxquels vous appartenez.

**groups** affiche les groupes auxquels vous appartenez

**w** liste les utilisateurs actuellement connectés sur la machine, et ce qu'ils font.

**last** affiche les dernières connexions

**getent passwd** affiche la liste des comptes utilisateurs présents sur le système.

# Utilisateurs: commandes associées

```
tom@cafeine:~$ groups
maison adm dialout cdrom audio src video plugdev netdev fuse scanner
tom@cafeine:~$
```

**id** affiche votre identifiant numérique et les groupes auxquels vous appartenez.

**groups** affiche les groupes auxquels vous appartenez

**w** liste les utilisateurs actuellement connectés sur la machine, et ce qu'ils font.

**last** affiche les dernières connexions

**getent passwd** affiche la liste des comptes utilisateurs présents sur le système.

# Utilisateurs: commandes associées

```
tom@cafeine:~$ w
12:01:58 up 9 days, 19:27, 3 users, load average: 0.06, 0.21, 0.24
USER      TTY      FROM          LOGIN@      IDLE        JCPU       PCPU       WHAT
tom       tty7     :0            Sat21       0.00s      9:30m     0.06s     /bin/bash /home/tom
tom       pts/5    :0.0         11:56       5:48m     0.22s     0.22s     zsh
tom       pts/2    :0.0         11:56       0.00s      1.38s     0.00s     w
tom@cafeine:~$
```

**id** affiche votre identifiant numérique et les groupes auxquels vous appartenez.

**groups** affiche les groupes auxquels vous appartenez

**w** liste les utilisateurs actuellement connectés sur la machine, et ce qu'ils font.

**last** affiche les dernières connexions

**getent passwd** affiche la liste des comptes utilisateurs présents sur le système.

## Utilisateurs: commandes associées

```

tom      tty1      Tue Jul 29 19:06 - 19:07 (00:01)
tom      tty1      Tue Jul 29 19:06 - 19:06 (00:00)
reboot   system boot 2.6.25-2-powerpc Tue Jan 5 08:05 - 19:30 (-11517+-20;
tom      tty7      Tue Jul 29 18:26 - crash (11517+21;57
reboot   system boot 2.6.25-2-powerpc Tue Jan 5 07:24 - 19:30 (-11517+-20;
tom      tty7      Tue Jul 29 17:50 - crash (11517+21;53
reboot   system boot 2.6.25-2-powerpc Tue Jan 5 06:49 - 19:30 (-11517+-19;
tom      tty7      Mon Jul 28 11:13 - crash (11519+03;54
reboot   system boot 2.6.25-2-powerpc Mon Jan 4 00:12 - 19:30 (-11516+-13;
tom      tty7      Sun Jul 27 20:56 - down (00:35)
reboot   system boot 2.6.25-2-powerpc Sun Jan 3 09:55 - 21:31 (-11517+-20;
reboot   system boot 2.6.25-2-powerpc Sun Jan 3 01:50 - 21:31 (-11517+-12;
reboot   system boot 2.6.25.6 Sun Jan 3 01:46 - 01:48 (00:01)
tom      tty7      Sun Jan 3 00:45 - down (-11517+-20;
reboot   system boot 2.6.25.6 Sun Jan 3 00:44 - 12:44 (-11517+-20;
reboot   system boot 2.6.25.6 Sat Jan 2 01:39 - 21:30 (-11522+-12;

```

**id** affiche votre identifiant numérique et les groupes auxquels vous appartenez.

**groups** affiche les groupes auxquels vous appartenez

**w** liste les utilisateurs actuellement connectés sur la machine, et ce qu'ils font.

**last** affiche les dernières connexions

**getent passwd** affiche la liste des comptes utilisateurs présents sur le système.

# Utilisateurs: commandes associées

```
www-data:x:33:33:www-data:/var/www:/bin/sh
backup:x:34:34:backup:/var/backups:/bin/sh
list:x:38:38:Mailng List Manager:/var/list:/bin/sh
irc:x:39:39:ircd:/var/run/ircd:/bin/sh
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/bin/sh
nobody:x:65534:65534:nobody:/nonexistent:/bin/sh
messagebus:x:101:103:/:/var/run/dbus:/bin/false
tom:x:1234:1234:Thomas Constans,,,:/home/tom:/bin/zsh
libuuid:x:102:104:/:/var/lib/libuuid:/bin/sh
statd:x:104:65534:/:/var/lib/nfs:/bin/false
```

**id** affiche votre identifiant numérique et les groupes auxquels vous appartenez.

**groups** affiche les groupes auxquels vous appartenez

**w** liste les utilisateurs actuellement connectés sur la machine, et ce qu'ils font.

**last** affiche les dernières connexions

**getent passwd** affiche la liste des comptes utilisateurs présents sur le système.

# Permissions de fichiers

- 1 Installation
- 2 Paramétrage de l'environnement
- 3 Vi
- 4 Gérer le système de fichiers
- 5 Traitements des fichiers textes
- 6 Gestion de Processus
- 7 Planification de tâches
- 8 Utilisateurs
- 9 Permissions de fichiers**
- 10 Système de fichiers
- 11 Arrêt et démarrage
- 12 Réseau TCP/IP sous GNU/Linux
- 13 Gestion des applications
- 14 Sch



# Permissions

Il y a 3 droits principaux, avec une valeur numérique et un symbole:

symbole	valeur	description

# Permissions

Il y a 3 droits principaux, avec une valeur numérique et un symbole:

symbole	valeur	description
r	4	lecture

# Permissions

Il y a 3 droits principaux, avec une valeur numérique et un symbole:

symbole	valeur	description
r	4	lecture
w	2	écriture

# Permissions

Il y a 3 droits principaux, avec une valeur numérique et un symbole:

symbole	valeur	description
r	4	lecture
w	2	écriture
x	1	exécution / accès

## Permissions sur les fichiers et les répertoires

Si elles sont les même, les permissions sur les fichiers et les répertoires n'ont pas le même effet:

permissions	fichier	répertoire
lecture	le contenu du fichier peut être lu	le contenu du répertoire peut être affiché
écriture	le fichier peut être modifié	les fichiers et répertoire présents dans le répertoire peuvent être <b>supprimés</b> .
exécution	le fichier peut être exécuté	on peut utiliser le répertoire comme répertoire courant.

# Permissions

Les permissions s'appliquent à **3** entités différentes:

**Propriétaire (u)** : l'utilisateur à l'origine du fichier

**Groupe Propriétaire (g)** : groupe principal auquel appartient le propriétaire du fichier

**Autres (o)**: le reste des utilisateurs

# Permissions

Les permissions s'appliquent à **3** entités différentes:

**Propriétaire (u)** : l'utilisateur à l'origine du fichier

**Groupe Propriétaire (g)** : groupe principal auquel appartient le propriétaire du fichier

**Autres (o)**: le reste des utilisateurs

```
tom@workine:~$ ls -l ~
drwx----- 2 tom maison 4,0K jui 23 08:33 PDF
drwxr-xr-x 2 tom maison 4,0K jan 9 2008 Templates
-rw-r--r-- 1 tom maison 8,0K jui 23 09:47 console.png
drwxr-xr-x 2 tom maison 4,0K jui 18 11:29 public_html
...
```

## Commandes associées

Les commandes suivantes permettent de modifier les permissions d'un fichier et de modifier son propriétaire:

**chmod mode fichier:** permet de modifier les permissions. *mode* peut être spécifié en mode numérique ou symbolique.

```
chmod 644 permissions.pdf
```

ou

```
chmod u+rw,g+r,o-r permissions.pdf
```



## Commandes associées

Les commandes suivantes permettent de modifier les permissions d'un fichier et de modifier son propriétaire:

**chmod mode fichier:** permet de modifier les permissions. *mode* peut être spécifié en mode numérique ou symbolique.

**chown user fichier:** rend *user* propriétaire du fichier.

```
chown tom permissions.pdf
```

## Commandes associées

Les commandes suivantes permettent de modifier les permissions d'un fichier et de modifier son propriétaire:

**chmod mode fichier:** permet de modifier les permissions. *mode* peut être spécifié en mode numérique ou symbolique.

**chown user fichier:** rend *user* propriétaire du fichier.

**chgrp groupe fichier:** rend le groupe *group* propriétaire du fichier

```
chgrp stagiaires permissions.pdf
```

## Déterminer les permissions à appliquer

L'utilisateur "*tom*" souhaite accorder les permissions suivantes sur le répertoire */home/tom/Exercices*:

- ▶ Tous les accès pour le propriétaire (lui-même)
- ▶ Accès en lecture pour les membres de son groupe principal
- ▶ Aucun accès pour les autres utilisateurs

## Déterminer les permissions à appliquer

L'utilisateur "tom" souhaite accorder les permissions suivantes sur le répertoire `/home/tom/Exercices`:

- ▶ Tous les accès pour le propriétaire (lui-même)
- ▶ Accès en lecture pour les membres de son groupe principal
- ▶ Aucun accès pour les autres utilisateurs

Valeur	User	Group	Other
<b>R</b> (4)			
<b>W</b> (2)			
<b>X</b> (1)			

## Déterminer les permissions à appliquer

L'utilisateur "tom" souhaite accorder les permissions suivantes sur le répertoire `/home/tom/Exercices`:

- ▶ Tous les accès pour le propriétaire (lui-même)
- ▶ Accès en lecture pour les membres de son groupe principal
- ▶ Aucun accès pour les autres utilisateurs

Valeur	User	Group	Other
<b>R(4)</b>	<b>4</b>		
<b>W(2)</b>	<b>2</b>		
<b>X(1)</b>	<b>1</b>		

## Déterminer les permissions à appliquer

L'utilisateur "tom" souhaite accorder les permissions suivantes sur le répertoire `/home/tom/Exercices`:

- ▶ Tous les accès pour le propriétaire (lui-même)
- ▶ Accès en lecture pour les membres de son groupe principal
- ▶ Aucun accès pour les autres utilisateurs

Valeur	User	Group	Other
<b>R(4)</b>	<b>4</b>	<b>4</b>	
<b>W(2)</b>	<b>2</b>	<b>0</b>	
<b>X(1)</b>	<b>1</b>	<b>1</b>	

## Déterminer les permissions à appliquer

L'utilisateur "tom" souhaite accorder les permissions suivantes sur le répertoire `/home/tom/Exercices`:

- ▶ Tous les accès pour le propriétaire (lui-même)
- ▶ Accès en lecture pour les membres de son groupe principal
- ▶ **Aucun accès pour les autres utilisateurs**

Valeur	User	Group	Other
<b>R(4)</b>	<b>4</b>	<b>4</b>	<b>0</b>
<b>W(2)</b>	<b>2</b>	<b>0</b>	<b>0</b>
<b>X(1)</b>	<b>1</b>	<b>1</b>	<b>0</b>

## Déterminer les permissions à appliquer

L'utilisateur "tom" souhaite accorder les permissions suivantes sur le répertoire `/home/tom/Exercices`:

- ▶ Tous les accès pour le propriétaire (lui-même)
- ▶ Accès en lecture pour les membres de son groupe principal
- ▶ Aucun accès pour les autres utilisateurs

Valeur	User	Group	Other
<b>R(4)</b>	<b>4</b>	<b>4</b>	<b>0</b>
<b>W(2)</b>	<b>2</b>	<b>0</b>	<b>0</b>
<b>X(1)</b>	<b>1</b>	<b>1</b>	<b>0</b>



## Déterminer les permissions à appliquer

L'utilisateur "tom" souhaite accorder les permissions suivantes sur le répertoire `/home/tom/Exercices`:

- ▶ Tous les accès pour le propriétaire (lui-même)
- ▶ Accès en lecture pour les membres de son groupe principal
- ▶ Aucun accès pour les autres utilisateurs

Valeur	User	Group	Other
<b>R(4)</b>	<b>4</b>	<b>4</b>	<b>0</b>
<b>W(2)</b>	<b>2</b>	<b>0</b>	<b>0</b>
<b>X(1)</b>	<b>1</b>	<b>1</b>	<b>0</b>
	7		

## Déterminer les permissions à appliquer

L'utilisateur "tom" souhaite accorder les permissions suivantes sur le répertoire `/home/tom/Exercices`:

- ▶ Tous les accès pour le propriétaire (lui-même)
- ▶ Accès en lecture pour les membres de son groupe principal
- ▶ Aucun accès pour les autres utilisateurs

Valeur	User	Group	Other
<b>R(4)</b>	<b>4</b>	<b>4</b>	<b>0</b>
<b>W(2)</b>	<b>2</b>	<b>0</b>	<b>0</b>
<b>X(1)</b>	<b>1</b>	<b>1</b>	<b>0</b>
	7	5	

## Déterminer les permissions à appliquer

L'utilisateur "tom" souhaite accorder les permissions suivantes sur le répertoire `/home/tom/Exercices`:

- ▶ Tous les accès pour le propriétaire (lui-même)
- ▶ Accès en lecture pour les membres de son groupe principal
- ▶ Aucun accès pour les autres utilisateurs

Valeur	User	Group	Other
<b>R(4)</b>	<b>4</b>	<b>4</b>	<b>0</b>
<b>W(2)</b>	<b>2</b>	<b>0</b>	<b>0</b>
<b>X(1)</b>	<b>1</b>	<b>1</b>	<b>0</b>
	7	5	0

Ce qui donne `chmod 750 /home/tom/Exercices`

# Les permissions spéciales

Elles sont généralement à 0, mais peuvent être utiles dans certains cas:

**La permission `Suid`:** permet d'exécuter une commande avec l'identité de son propriétaire.

**La permission `sGid`:** permet d'exécuter une commande avec l'identité de son groupe propriétaire.

**La permission `sGid`** , appliquée sur un *répertoire*, rend le groupe propriétaire du répertoire *propriétaire* des fichiers et répertoires qui y seront créés.

**La permission `sTicky`** , appliquée sur un répertoire, empêche les utilisateurs de pouvoir supprimer les fichiers dont ils ne sont pas propriétaires.

# Les permissions spéciales

Le calcul de ces permissions est similaire à celui des permissions standards.

Valeur		<b>U</b> ser	<b>G</b> roup	<b>O</b> ther
4				
2				
1				

# Les permissions spéciales

Le calcul de ces permissions est similaire à celui des permissions standards.

Valeur	spec	<b>U</b> ser	<b>G</b> roup	<b>O</b> ther
4		r	r	-
2		w	-	-
1		x	x	-

# Les permissions spéciales

Le calcul de ces permissions est similaire à celui des permissions standards.

Valeur	spec	<b>U</b> ser	<b>G</b> roup	<b>O</b> ther
4	Suid	r	r	-
2	sGid	w	-	-
1		x	x	-

# Les permissions spéciales

Le calcul de ces permissions est similaire à celui des permissions standards.

Valeur	spec	<b>U</b> ser	<b>G</b> roup	<b>O</b> ther
4	Suid	r	r	-
2	sGid	w	-	-
1	sTicky	x	x	-



# Les permissions spéciales

Le calcul de ces permissions est similaire à celui des permissions standards.

Valeur	spec	User	Group	Other
4	Suid	r	r	-
2	sGid	w	-	-
1	sTicky	x	x	-

Si on veut appliquer la permissions *sTicky* au répertoire */home/tom/Exercices* on aura:

# Les permissions spéciales

Le calcul de ces permissions est similaire à celui des permissions standards.

Valeur	spec	User	Group	Other
4	Suid	r	r	-
2	sGid	w	-	-
1	sTicky	x	x	-

Si on veut appliquer la permissions *sTicky* au répertoire */home/tom/Exercices* on aura:

```
chmod 1750 /home/tom/Exercices
```

# Les permissions par défaut

Les permissions affectées à un nouveau répertoire ou fichier dépendent de la valeur de *l'umask*.

- ▶ Pour un répertoire, les droits par défaut sont à 0777 - l'umask du créateur.
- ▶ Pour un fichier, ses droits sont à 0666 - l'umask du créateur.

Ainsi, avec un *umask* à 0027, un nouveau fichier aura pour permissions:

## Les permissions par défaut

Les permissions affectées à un nouveau répertoire ou fichier dépendent de la valeur de *l'umask*.

- ▶ Pour un répertoire, les droits par défaut sont à 0777 - l'umask du créateur.
- ▶ Pour un fichier, ses droits sont à 0666 - l'umask du créateur.

Ainsi, avec un *umask* à 0027, un nouveau fichier aura pour permissions:  
0666 permission de départ

## Les permissions par défaut

Les permissions affectées à un nouveau répertoire ou fichier dépendent de la valeur de *l'umask*.

- ▶ Pour un répertoire, les droits par défaut sont à 0777 - l'umask du créateur.
- ▶ Pour un fichier, ses droits sont à 0666 - l'umask du créateur.

Ainsi, avec un *umask* à 0027, un nouveau fichier aura pour permissions:  
0666 permission de départ  
-0027 umask

## Les permissions par défaut

Les permissions affectées à un nouveau répertoire ou fichier dépendent de la valeur de *l'umask*.

- ▶ Pour un répertoire, les droits par défaut sont à 0777 - l'umask du créateur.
- ▶ Pour un fichier, ses droits sont à 0666 - l'umask du créateur.

Ainsi, avec un *umask* à 0027, un nouveau fichier aura pour permissions:

0666 permission de départ

-0027 umask

=0640 permissions effectives

## En résumé

Pour mettre en place les permissions souhaitées:

- ▶ On joue sur l'utilisateur et/ou le groupe propriétaire du fichier

## En résumé

Pour mettre en place les permissions souhaitées:

- ▶ On joue sur l'utilisateur et/ou le groupe propriétaire du fichier (à l'aide des commandes `chown` et `chgrp`)
- ▶ On applique à ces propriétaires les permissions



## En résumé

Pour mettre en place les permissions souhaitées:

- ▶ On joue sur l'utilisateur et/ou le groupe propriétaire du fichier (à l'aide des commandes `chown` et `chgrp`)
- ▶ On applique à ces propriétaires les permissions

## En résumé

Pour mettre en place les permissions souhaitées:

- ▶ On joue sur l'utilisateur et/ou le groupe propriétaire du fichier (à l'aide des commandes `chown` et `chgrp`)
- ▶ On applique à ces propriétaires les permissions (à l'aide de `chmod`)

# Système de fichiers

- 1 Installation
- 2 Paramétrage de l'environnement
- 3 Vi
- 4 Gérer le système de fichiers
- 5 Traitements des fichiers textes
- 6 Gestion de Processus
- 7 Planification de tâches
- 8 Utilisateurs
- 9 Permissions de fichiers
- 10** **Système de fichiers**
  - Introduction
  - Fichiers de périphérique
  - Partitions Linux
  - Systèmes de fichier
  - Montage
- 11 Arrêt et démarrage
- 12 Réseau TCP/IP sous

# Partitions

- ▶ Une *partition* est une subdivision d'un périphérique de stockage, destinée à recevoir un système de fichier.
- ▶ La définition du partitionnement d'un disque est une étape essentielle du processus d'installation d'un système Linux.
- ▶ Le système de fichier d'une partition est rendu disponible par l'opération de "montage".
- ▶ Le point d'entrée (de montage) d'une partition est un répertoire.

# Création d'une partition

Par la commande `fdisk fichier_periph_disque`.

- ▶ Il existe aussi le programme “graphique” `cdisk`.
- ▶ La structure du disque n'est pas modifiée tant que les changements n'ont pas été écrits (**W**).
- ▶ Toute modification apportée à la structure d'un disque rendra les données existantes difficilement accessibles. . .

# Fichiers de périphérique

Sous linux, “tout est fichier”.

- ▶ Disques et partitions n'échappent pas à la règle:
- ▶ Les disques scsi, sata et usb sont gérés par le driver *sd*, d'où le fichier de périphérique correspondant: */dev/sd* suivi d'une lettre.
- ▶ Les disques IDE sont référencés par les fichiers */dev/hd*, suivi d'une lettre.
- ▶ Les partitions sont représentés par leur fichier de périphérique disque, suffixée d'une ° de partition.
- ▶ Ex:
  - /dev/hdc* correspond au disque esclace du 2<sup>ième</sup> port ide.
  - /dev/sdb3* représente la 3<sup>ième</sup> partition du 2<sup>ième</sup> disque usb, sata ou scsi du système.

# Identification des périphériques

Lors de l'insertion d'un nouveau périphérique de stockage, celui-ci se voit affecter un fichier correspondant.

- ▶ Tout l'art est d'identifier ce fichier.
- ▶ Cela peut se faire en examinant les logs système

## Identification des périphériques

```
scsi 4:0:0:0: Direct-Access      LEXAR    DIGIT  
AL FILM      /W1. PQ: 0 ANSI: 2  
sd 4:0:0:0: [sdb] 125952 512-byte hardware sec  
tors (64 MB)  
  
sd 4:0:0:0: [sdb] Assuming drive cache: write  
through  
sdb: sdb1  
sd 4:0:0:0: [sdb] Attached SCSI removable disk  
usb-storage: device scan complete
```

Lors de l'insertion d'un nouveau périphérique de stockage, celui-ci se voit affecter un fichier correspondant.

- ▶ Tout l'art est d'identifier ce fichier.
- ▶ Cela peut se faire en examinant les logs système
- ▶ Et notamment le résultat de la commande *dmesg*



# lsblk

La commande `lsblk` permet également de lister les différents périphériques de stockage disponible sur le système

```
[root@localhost:~] # lsblk
NAME                                MAJ:MIN RM  SIZE RO TYPE
MOUNTPOINT
sda                                  8:0    0   8G  0 disk
sda1                                8:1    0 500M  0 part /
boot
sda2                                8:2    0 7,5G  0 part
centos-root 253:0    0 6,7G  0 lvm  /
centos-swap 253:1    0 820M  0 lvm  [
SWAP]
sdb                                  8:16    0   2G  0 disk
sdb1                                8:17    0   2G  0 part
sr0                                  11:0    1 1024M  0 rom
```

# Partitions

GNU/Linux demande un minimum de 2 partitions:

- 1 partition de swap, utilisé pour la mémoire virtuelle.
- 2 partition “racine”.

# Partitions

GNU/Linux demande un minimum de 2 partitions:

- 1 partition de swap, utilisé pour la mémoire virtuelle.
- 2 partition “racine”.

Il est pratique de dédier une partition pour les répertoires suivants:

`/home` dans le cas d'un serveur de fichiers.

`/var`

`/tmp`

## En effet

- ▶ Un système Linux *n'aime pas du tout* se retrouver sans espace disque disponible.
- ▶ Les répertoires dont la taille est variable et non maîtrisable mérite donc une partition dédiée.
- ▶ Ce qui limite les dégats.
- ▶ Mais fige la structure des systèmes de fichiers
- ▶ et rend difficile toute modification ultérieure.

# Systèmes de fichier

Les systèmes de fichiers permettent d'organiser les données sur un espace de stockage.

- ▶ Généralement en les organisant en répertoires.
- ▶ Et en proposant différents attributs et caractéristiques (nom, permissions, etc).

# Ext3

- ▶ Le principal système de fichier sous linux
- ▶ Bon compromis entre performance et stabilité.
- ▶ Journalisation => récupération rapide.

# Création d'un système de fichier

par la commande `mkfs.ext3 /dev/sda1`

- ▶ Les caractéristiques du système de fichier peuvent être lues/modifiées par la commande `tune2fs`.
- ▶ Utile pour, notamment, modifier les modalités de vérifications automatiques, la quantité d'espace disque réservés, les algorithmes de stockage, etc.

# Montage

Le *montage* consiste à rendre accessible le contenu d'un système de fichier par l'intermédiaire d'un répertoire.

- ▶ Ce répertoire s'appelle le *point de montage* du système de fichier.
- ▶ Le montage peut se faire manuellement, pour les périphériques amovibles.
- ▶ ou automatiquement pour les périphériques système.
- ▶ un certain nombre d'utilitaires ont été créés pour monter automatiquement un périphérique lors de son insertion (cd, clé usb, ...).



# Montage

Le *montage* consiste à rendre accessible le contenu d'un système de fichier par l'intermédiaire d'un répertoire.

- ▶ Ce répertoire s'appelle le *point de montage* du système de fichier.
- ▶ Le montage peut se faire manuellement, pour les périphériques amovibles.
- ▶ ou automatiquement pour les périphériques système.
- ▶ un certain nombre d'utilitaires ont été créés pour monter automatiquement un périphérique lors de son insertion (cd, clé usb, ...).

Ex de montage manuel: `mount -t vfat /dev/sda1 /media/usb`

## fstab et montage auto

Le fichier `/etc/fstab` définit les systèmes de fichiers dont le montage est nécessaire au bon fonctionnement du système.

- ▶ Il est lu au démarrage
- ▶ Il contient les informations suivantes, en ligne:

periphérique

point de montage

type du système de fichier (ext3, vfat, auto, ...)

options de montage (defaults, (no)auto, user, acl, ...)

dump rarement utilisé, généralement 0

fsck indique l'ordre dans lequel les systèmes de fichier seront vérifiés au démarrage. root -> 1 . pas de vérification -> 0.

## fstab et montage auto

Le fichier `/etc/fstab` définit les systèmes de fichiers dont le montage est nécessaire au bon fonctionnement du système.

- ▶ Il est lu au démarrage
- ▶ Il contient les informations suivantes, en ligne:

periphérique

point de montage

type du système de fichier (ext3, vfat, auto, ...)

options de montage (defaults, (no)auto, user, acl, ...)

dump rarement utilisé, généralement 0

fsck indique l'ordre dans lequel les systèmes de fichier seront vérifiés au démarrage. root -> 1 . pas de vérification -> 0.

Si on appelle la commande `mount` avec 1 seul paramètre (point de montage ou périphérique), elle recherche les autres informations correspondantes dans le fichier `/etc/fstab`.

# Démontage

Il est obligatoire de démonter un système de fichier après utilisation.

- ▶ par la commande *umount point\_de\_montage*
- ▶ les périphériques listés comme “auto” dans */etc/fstab* sont automatiquement démontés avant l’arrêt de la machine.
- ▶ un périphérique utilisé ne peut pas être démonté
- ▶ La commande *lsof* aide à détecter les processus utilisant un périphérique.

# Les autres systèmes de fichier

**tmpfs** système de fichier en ram

**usbfs** pseudo système de fichier, représentant les périphériques usb du système.

**sysfs** contient une représentation des pilotes présents sur le système, et de leurs attributs

**proc** image / information du matériel et des processus

**devpts** utilisé pour les consoles virtuelles

...

# Taille des systèmes de fichier

La commande `df` permet d'obtenir cette taille

# Arrêt et démarrage

- 1 Installation
- 2 Paramétrage de l'environnement
- 3 Vi
- 4 Gérer le système de fichiers
- 5 Traitements des fichiers textes
- 6 Gestion de Processus
- 7 Planification de tâches
- 8 Utilisateurs
- 9 Permissions de fichiers
- 10 Système de fichiers
- 11 Arrêt et démarrage**
  - Les grandes étapes du démarrage
  - Systemd
  - Gestion des services
  - Unit-files personnalisées
- 12 Réseau TCP/IP sous

# Les grandes étapes

Un système GNU/Linux passe par les étapes suivantes lors du démarrage:

- 1 Bios
- 2 Chargeur de démarrage
- 3 Noyau
- 4 Détection et activation du périphérique “*racine*”.
- 5 Lancement du processus *systemd*.
- 6 Exécution des unités *systemd*
- 7 Lancement de l'utilitaire de connexion.



# Le chargeur de démarrage

Le rôle du chargeur de démarrage est de *détecter* et de *charger* le noyau.

- ▶ Il y a 2 gestionnaire de démarrage: 1 rudimentaire stocké dans le *MBR*, un autre stocké situé sur une partition du disque.
- ▶ Sous Linux, le chargeur de démarrage est *grub*
- ▶ Ils proposent un menu interactif permettant de choisir les différents noyaux présents sur le système.
- ▶ Une erreur de configuration peut entraîner un problème de démarrage.

# Grub

## Le **G**rand **U**nified **B**oot loader

- ▶ Son fichier de configuration est */boot/grub/menu.lst*
- ▶ Mini shell
- ▶ Pas de réinstallation suite à modification de configuration
- ▶ Plus souple d'utilisation

# Le noyau

Le noyau est le premier processus (dont le pid est 0) du système.

- ▶ Il initialise les principaux périphériques en chargeant les pilotes associés.
- ▶ Le noyau détecte et rend accessible le *système de fichier racine*.
- ▶ Il lance un certain nombre de processus chargés de gérer certaines parties du système ([kswapd], [kjournal], ...).
- ▶ Enfin, il exécute le 1er processus utilisateur: *init*.

# Systemd

*Systemd* a remplacé, sur les distributions récentes *init*. Il s'occupe de la configuration de la machine (paramétrage réseau, nom de machine, montage des systèmes de fichiers, etc.) et du lancement des différents services. Cela se fait au moyen de fichiers d'unités, ou "*unit files*" qui ont remplacé les anciens script d'*init*

# Les unit files

Les unit files sont de différent type (suivant l'extension) et offrent différentes fonctions.

les plus utilisées sont les units de type service.

La tendance actuelle est à la convergence de la configuration d'un système (montage, tâches planifiées, ...) via systemd.

Les unit-files par défaut des applications et services sont dans `/usr/lib/systemd/system/`.

Les unit-files personnalisées sont à mettre dans `/etc/systemd/system`.

À l'installation d'une nouvelle unit-file, il faut recharger systemd:  
`systemctl daemon-reload`.

# La gestion des services

Les services se gèrent de la manière suivante:

**arrêt et redémarrage** `systemctl start|stop|restart unit-file-name.service - ex`  
`systemctl restart httpd`

**Activation et désactivation** `systemctl enable|disable unit-file-name.service - ex`  
`systemctl enable httpd`

## Unit-files personnalisées

Créer un fichier `/etc/systemd/system/foo.service` :

```
[Unit]
Description=script de démarrage foo
After=network.target remote-fs.target nss-
      lookup.target

[Service]
Type=notify
WorkingDirectory=/opt/bin
ExecStart=/opt/bin/foo start

[Install]
WantedBy=multi-user.target
```

Ne pas hésiter à consulter les pages de man *systemd.service* et *systemd.unit*

# Commandes associées

`systemctl` sans argument, liste toutes les unités chargées, y compris celles en **échec**

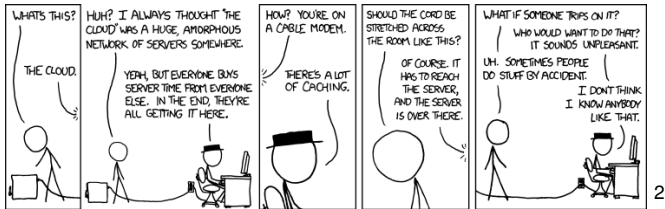
`systemd-analyze` permet d'identifier les unités qui ralentissent le démarrage du système



# Réseau TCP/IP sous GNU/Linux

- 1 Installation
- 2 Paramétrage de l'environnement
- 3 Vi
- 4 Gérer le système de fichiers
- 5 Traitements des fichiers textes
- 6 Gestion de Processus
- 7 Planification de tâches
- 8 Utilisateurs
- 9 Permissions de fichiers
- 10 Système de fichiers
- 11 Arrêt et démarrage
- 12 Réseau TCP/IP sous GNU/Linux**
  - Configuration TCP/IP
  - Parefeu
- 13 Gestion des applications

# Linux



# Nom de machine

Modifié par la commande *hostnamectl*.

`hostnamectl` affiche le nom actuel.

`hostnamectl set-hostname nom` modifie le nom de la machine.

## Vérification de la configuration

La configuration des interfaces est visible par la commande `ip a`

```
tom@cafeine$ /sbin/ifconfig eth2
eth2      Link encap:Ethernet  HWaddr 00:14:51:db:cc:ee

          inet adr:192.168.10.102  Bcast:192.168.10.255
          Masque:255.255.255.0
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
```

```
ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536
   qdisc noqueue state UNKNOWN group
   default qlen 1000
   link/loopback 00:00:00:00:00:00 brd
     00:00:00:00:00:00
   inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft
       forever
   inet6 ::1/128 scope host
       valid_lft forever preferred_lft
```

# Configuration des interfaces

La gestion du réseau sur les distributions récentes est assurée par *NetworkManager*. On ne touche plus aux fichiers de configuration, on passe désormais par différentes interfaces:

- 1 l'interface graphique, si on est en environnement graphique
- 2 l'interface *nmtui* (Network Manager Text User Interface)
- 3 ou l'interface *nmcli*

Cette dernière commande va principalement servir à lister, activer ou désactiver une connexion:

- ▶ `nmcli c`
- ▶ `nmcli c up enp0s3`
- ▶ `nmcli c down enp0s3`

# Configuration de la résolution locale

```
cat /etc/hosts
127.0.0.1    localhost localhost.
             localdomain localhost4 localhost4.
             localdomain4 cafeine foobar
#:::1      localhost localhost.
             localdomain localhost6 localhost6.
             localdomain6
192.168.10.96 cafeine
192.168.10.99 turbine
```

# Configuration des serveurs de noms

Les adresses des serveurs de nom sont renseignées dans le fichier `/etc/resolv.conf`:

```
search opendoor.fr
nameserver 192.168.10.254
```

**nameserver** permet d'indiquer l'adresse du serveur DNS. il peut y en avoir plusieurs (3).

**domain** indique le nom de domaine local.

**search** liste des domaines qui seront accolés au critère de recherche, si celui-ci n'est pas un fqdn.

**options** permet de modifier le comportement du résolveur

Les directives *domain* et *search* sont mutuellement exclusives.

# Configuration du routage

Le routage indique à une machine ses possibilités d'accès sur l'extérieur. On distingue 3 types:

**minimal:** lors de la configuration des interfaces, une route est automatiquement créée vers le réseau local.

**statique:** Les tables de routages sont maintenues manuellement.

**dynamique:** Des protocoles spécifiques maintiennent à jour les tables de routage.



## Affichage de la table de routage

La commande `ip r(oute)` permet d'afficher la table de routage locale.

```
# ip r
default via 10.251.255.254 dev wlp3s0
    proto dhcp metric 20600
10.251.0.0/16 dev wlp3s0 proto kernel
    scope link src 10.251.150.229 metric
    600
100.0.0.0/8 dev vboxnet0 proto kernel
    scope link src 100.0.0.1
172.17.0.0/16 dev docker0 proto kernel
    scope link src 172.17.0
```

# Table de routage

Les informations renvoyées par la commande *ip r* sont:

**default ou réseau:** passerelle par défaut ou réseau de destination.

**Passerelle:** adresse du routeur.

**dev wlp3s0:** périphérique à utiliser

**Metric:** *coût* de la route.

## Ajout et modification d'une route

La table de routage peut être modifiée lors de l'activation d'une interface, par l'intermédiaire de la directive *gateway* du fichier */etc/network/interfaces*.

La commande *ip r(oute) add* permet de modifier la table de routage:

`ip r add 10/8 dev eth0` ajout de la route vers le réseau local

`ip r add default via 10.0.0.254` ajout de la passerelle par défaut.

`ip r del default` suppression de la route par défaut.

# Parefeu

S'il s'appuie toujours sur netfilter/iptables, le parefeu sur RedHat CentOS est désormais géré par *firewalld*.

# Zones

Les zones firewalld

# Gestion des applications

- 1 Installation
- 2 Paramétrage de l'environnement
- 3 Vi
- 4 Gérer le système de fichiers
- 5 Traitements des fichiers textes
- 6 Gestion de Processus
- 7 Planification de tâches
- 8 Utilisateurs
- 9 Permissions de fichiers
- 10 Système de fichiers
- 11 Arrêt et démarrage
- 12 Réseau TCP/IP sous GNU/Linux
- 13 Gestion des applications**
  - Introduction
  - Dépendances

## INSTALL.SH

```
#!/bin/bash
```

```
pip install "$1" &  
easy_install "$1" &  
brew install "$1" &  
npm install "$1" &  
yum install "$1" & dnf install "$1" &  
docker run "$1" &  
pkg install "$1" &  
apt-get install "$1" &  
sudo apt-get install "$1" &  
steamcmd +app_update "$1" validate &  
git clone https://github.com/"$1"/"$1" &  
cd "$1"; ./configure; make; make install &
```

# Introduction

Pour rappel, une *distribution* GNU/Linux propose

- ▶ Un noyau Linux
- ▶ Un ensemble d'utilitaire et de services (ex, support, media d'installation, mises à jour, ...)
- ▶ Une collection de logiciels intégrée au système



# Bibliothèques partagées et dépendances

Application a

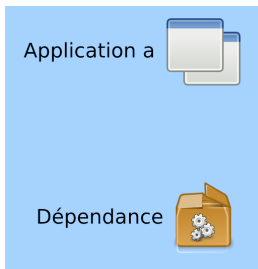


Dépendance



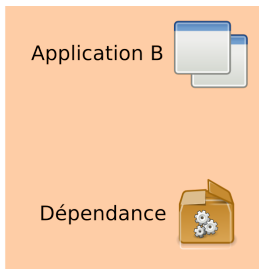
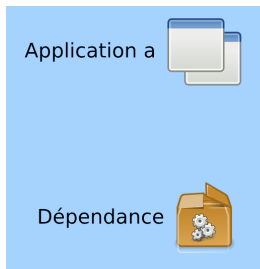
- ▶ 1 logiciel nécessite une fonction proposée par le paquet logiciel “dépendance”.
- ▶ La 1<sup>ère</sup> solution est de packager les 2 ensemble.
- ▶ Si un 2<sup>ème</sup> logiciel demande la même *dépendance*, cette solution devient suboptimale.
- ▶ Les logiciels GNU/Linux sont basés sur le principe de la *bibliothèque de fonctions partagées*.
- ▶ “*dépendance*” est packagée indépendamment. On obtient un système modulaire.

# Bibliothèques partagées et dépendances



- ▶ 1 logiciel nécessite une fonction proposée par le paquet logiciel “dépendance”.
- ▶ La 1<sup>ière</sup> solution est de packager les 2 ensemble.
- ▶ Si un 2<sup>ième</sup> logiciel demande la même *dépendance*, cette solution devient suboptimale.
- ▶ Les logiciels GNU/Linux sont basés sur le principe de la *bibliothèque de fonctions partagées*.
- ▶ “*dépendance*” est packagée indépendamment. On obtient un système modulaire.

# Bibliothèques partagées et dépendances



- ▶ 1 logiciel nécessite une fonction proposée par le paquet logiciel “dépendance”.
- ▶ La 1<sup>ère</sup> solution est de packager les 2 *ensemble*.
- ▶ Si un 2<sup>ème</sup> logiciel demande la même *dépendance*, cette solution devient *suboptimale*.
- ▶ Les logiciels GNU/Linux sont basés sur le principe de la *bibliothèque de fonctions partagées*.
- ▶ “*dépendance*” est packagée indépendamment. On obtient un système modulaire.

# Bibliothèques partagées et dépendances

Application a



Application B

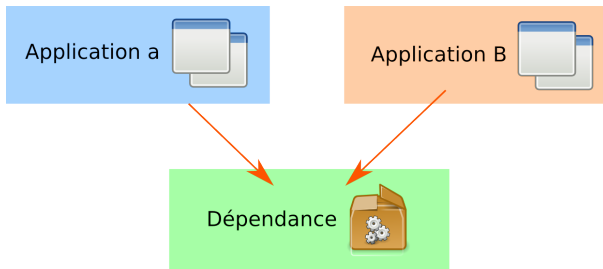


Dépendance



- ▶ 1 logiciel nécessite une fonction proposée par le paquet logiciel “dépendance”.
- ▶ La 1<sup>ère</sup> solution est de packager les 2 *ensemble*.
- ▶ Si un 2<sup>ème</sup> logiciel demande la même *dépendance*, cette solution devient suboptimale.
- ▶ Les logiciels GNU/Linux sont basés sur le principe de la *bibliothèque de fonctions partagées*.
- ▶ “*dépendance*” est packagée indépendamment. On obtient un système modulaire.

# Bibliothèques partagées et dépendances



- ▶ 1 logiciel nécessite une fonction proposée par le paquet logiciel “dépendance”.
- ▶ La 1<sup>ère</sup> solution est de packager les 2 *ensemble*.
- ▶ Si un 2<sup>ème</sup> logiciel demande la même *dépendance*, cette solution devient suboptimale.
- ▶ Les logiciels GNU/Linux sont basés sur le principe de la *bibliothèque de fonctions partagées*.
- ▶ “*dépendance*” est packagée indépendamment. On obtient un système modulaire.

# Question

## Question:

Comment s'assurer qu'une application sera installée avec toutes ses dépendances

# Les gestionnaires de paquets

Les “package manager” sont une des particularités de chaque distribution.  
Leur rôle est:

- Installer** les logiciels
- Résoudre** les dépendances
- Désinstaller** les logiciels et leurs dépendances inutilisées.
- Maintenir** le système à jour.
- Rechercher** des logiciels

# Les gestionnaires de paquets

Les “package manager” sont une des particularités de chaque distribution.  
Leur rôle est:

**Installer** les logiciels

**Résoudre** les dépendances

**Désinstaller** les logiciels et leurs dépendances inutilisées.

**Maintenir** le système à jour.

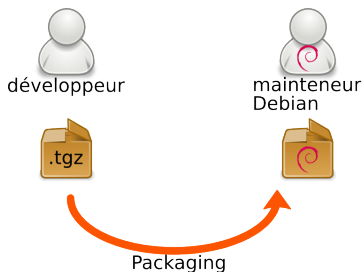
**Rechercher** des logiciels

*Apt* est le gestionnaire de paquet de GNU/Debian.

*Yum* est celui des distributions RedHat et CentOS

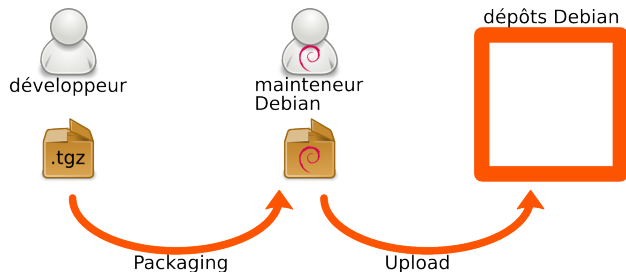


# les paquets



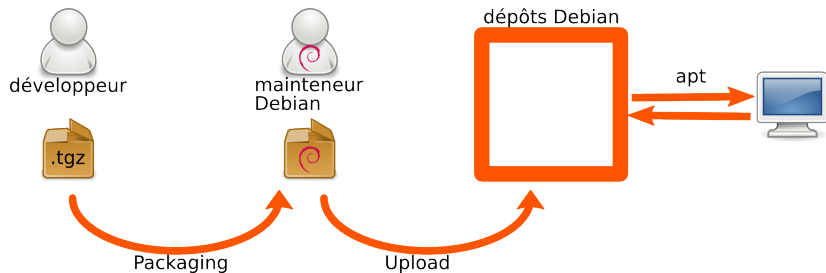
- ▶ Un logiciel (libre) est “*empaqueté*” au format “rpm”
- ▶ Il est ensuite inclu dans les dépôts CentOS/Debian
- ▶ Et devient donc disponible par l'intermédiaire de *yum*.

# les paquets



- ▶ Un logiciel (libre) est "*empaqueté*" au format "rpm"
- ▶ Il est ensuite inclu dans les dépôts CentOS/Debian
- ▶ Et devient donc disponible par l'intermédiaire de *yum*.

# les paquets



- ▶ Un logiciel (libre) est “*empaqueté*” au format “rpm”
- ▶ Il est ensuite inclu dans les dépôts CentOS/Debian
- ▶ Et devient donc disponible par l'intermédiaire de *yum*.

# Le format .rpm

un paquet debian comprend:

**Un en-tête** , qui contient des informations sur le paquet

**Une archive** , qui sera extraite lors de l'installation.

**Des scripts** , qui seront exécutés lors de l'installation et la désinstallation

## Configuration des dépôts

La connexion entre notre gestionnaire de paquet et les dépôts se fait via les fichiers de `/etc/yum.repos.d`. Configurer un nouveau dépôt revient à créer un nouveau fichier de définition dans ce répertoire. Cela peut se faire en installant un paquet. Voir par exemple le paquet `epel-release`.

Attention, les paquets d'un dépôt sont généralement signés. Si on configure un nouveau dépôt, il ne faut pas oublier la clé correspondante, sinon yum ne pourra pas vérifier l'intégrité des paquets lors de leur installation.

# Interrogation de la base des paquets locaux

la commande *rpm* va principalement être utilisé pour interroger la base des paquets installés:

`rpm -qa` liste de tous les paquets installés.

`rpm -qi paquet` affiche des informations sur un paquet.

`rpm -ql paquet` affiche le contenu d'un paquet.

# Gestion de paquets avec yum

**yum install package:** installe *package* et ses dépendances

**yum search motif** recherche un package dont la description ou le résumé contient "*motif*".

**yum remove package:** désinstalle *package* et ses dépendances

**yum update:** installation des dernières mises à jour disponibles.

**yum provides /chemin:** cherche les paquets contenant */chemin*

## Gestion de l'historique

La sous-commande *history* de yum permet de revoir et au besoin d'annuler toutes les commandes yum.

**yum history list** - affiche l'historique des installations, déinstallation, mise à jour de paquets

**yum history info 234** affiche des infos (principalement la liste des paquets concernés) par l'opération 234

**yum history undo 234** annule les actions réalisées par l'opération 234 on désinstalle ce qui aura été installé)

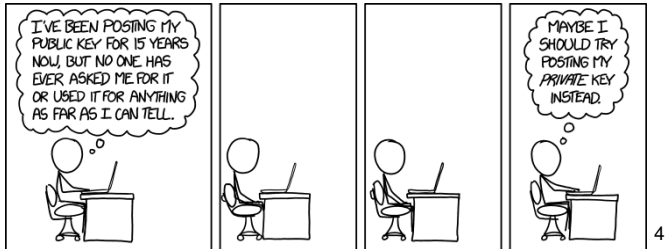
**yum history rollback 234** on annule toutes les opérations de la plus récente jusqu'à (y compris) l'opération 234.



# Ssh

- 1 Installation
- 2 Paramétrage de l'environnement
- 3 Vi
- 4 Gérer le système de fichiers
- 5 Traitements des fichiers textes
- 6 Gestion de Processus
- 7 Planification de tâches
- 8 Utilisateurs
- 9 Permissions de fichiers
- 10 Système de fichiers
- 11 Arrêt et démarrage
- 12 Réseau TCP/IP sous GNU/Linux
- 13 Gestion des applications
- 14 Ssh

# Linux



# Introduction

- ▶ *Ssh* est un protocole permettant de sécuriser une communication réseau.
- ▶ Il a été conçu pour remplacer *telnet* et *rlogin*.
- ▶ Il propose un mécanisme d'authentification des machines par *clé*.
- ▶ Il permet notamment:

# Introduction

- ▶ *Ssh* est un protocole permettant de sécuriser une communication réseau.
- ▶ Il a été conçu pour remplacer *telnet* et *rlogin*.
- ▶ Il propose un mécanisme d'authentification des machines par *clé*.
- ▶ Il permet notamment:
  - ▶ d'exécuter des commandes sur une machine distante.
  - ▶ de faire du *port-forwarding*.
  - ▶ de transférer des fichiers.

# Principes de fonctionnement

- ▶ Chaque machine dispose d'une paire de clé publique / privée.
- ▶ Une connexion ssh n'est possible qu'entre machines se "connaissant".

```
tom@cafeine:~$ ssh workine
The authenticity of host '[workine]:2222 ([192.168.10.2]:2222)' can't be established.
RSA key fingerprint is 22:fc:a4:6a:87:42:62:3b:de:d3:66:a2:3f:4c:bf:e4.
Are you sure you want to continue connecting (yes/no)? 
```

Une fois le canal établi, le système distant authentifie l'utilisateur.

- ▶ Soit par mot de passe.
- ▶ Soit par clé publique / privé.

# Installation et configuration

Le serveur ssh fait partie de l'installation de base et est généralement à l'écoute sur le port 22. Le client est également inclus dans l'installation de base.

```
workine:~# netstat -taupen |grep ssh
tcp6      0      0  :::22          :::*           LISTEN     0          25156105      27721/sshd
workine:~#
```

Ce daemon est géré par le fichier d'unité *sshd*.

## Cas d'utilisation

```
tom@cafeine$ ssh workine
tom@workine's password:
Linux workine 2.6.25-2-686 #1 SMP Fri Jun 27 0
3:23:20 UTC 2008 i686

The programs included with the Debian GNU/Linu
x system are free software;
the exact distribution terms for each program
are described in the
individual files in /usr/share/doc/*/copyright
.

Debian GNU/Linux comes with ABSOLUTELY NO WARR
ANTY, to the extent
permitted by applicable law.
Last login: Sun Aug 31 13:33:40 2008 from 192.
168.10.102
(0) (tom@workine)
```

- ▶ Shell sur une machine distante

# Cas d'utilisation

```
tom@cafeine$  
tom@cafeine$ ssh tconstans@opendoor.fr ls -al  
~
```

- ▶ Shell sur une machine distante
- ▶ Exécution d'une commande à distance



# Cas d'utilisation

```
(47) (tom@cafeine)ssh -L 1234:serveur:80 radm@passerelle.secure.org
```

- ▶ Shell sur une machine distante
- ▶ Exécution d'une commande à distance
- ▶ Mise en place d'un tunnel ssh

# Cas d'utilisation

```
tom@cafeine$ scp formation.pdf supports.kiodan  
.fr:/var/www/html/Supports/
```

- ▶ Shell sur une machine distante
- ▶ Exécution d'une commande à distance
- ▶ Mise en place d'un tunnel ssh
- ▶ Copie de fichier sécurisée.

## Configuration du serveur

La configuration est localisée dans le fichier `/etc/ssh/sshd_config`. Les principales directives sont:

**Port** et **ListenAddress** paramètre réseau.

**PubKeyAuthentication** activation de l'authentification par clé publique/privée.

**PasswordAuthentication** activation de l'authentification par mot de passe.

**AllowUsers** liste des utilisateurs autorisés à se connecter par ssh.

**AllowGroups** liste des groupes autorisés.

**Match** configuration conditionnelle.

**PermitRootLogin** "no", "forced-commands-only", "without-password"

**Commands** liste des commandes autorisées.

**ClientAliveCountMax** et

**ClientAliveInterval** gestion des sessions inactives.

# Principe

L'authentification *utilisateur* par clé est plus simple et plus sécurisée:

- ▶ Ni la clé privée, ni la “*passphrase*” ne sont transmises sur le réseau.
- ▶ 1 seul mot de passe à connaître: celui qui protège la clé privée.
- ▶ L'utilisateur est authentifié si sa clé publique est connue du serveur.



La clé publique de bob  
n'est pas connue du  
serveur

La connexion n'est pas  
acceptée.



La clé publique de bob  
est connue du serveur

La connexion est  
acceptée.

# Principe

Si vous n'êtes pas dans la liste des invités, vous restez à la porte



# Mise en oeuvre

**ssh-keygen:** génération d'une paire de clé publique / privée

**ssh-copy-id:** copie de la clé publique sur le serveur distant.

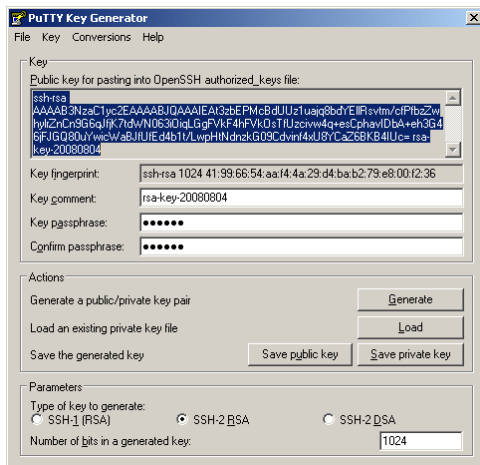
**ssh:** utilisation:

- ▶ déverrouillage de la clé privée à l'aide de la "*passphrase*" associée.
- ▶ session ssh

# Cache

- ▶ ssh-agent permet de mettre en cache, pendant une durée déterminée, la version en clair de sa clé privée.
- ▶ Cela autorise de multiples connexion ssh, en n'entrant la "*passphrase*" qu'une seule fois.
- ▶ ssh-agent -t 600
- ▶ ssh-add
- ▶ ssh-add -L
- ▶ ssh-add -D
- ▶ ssh -A

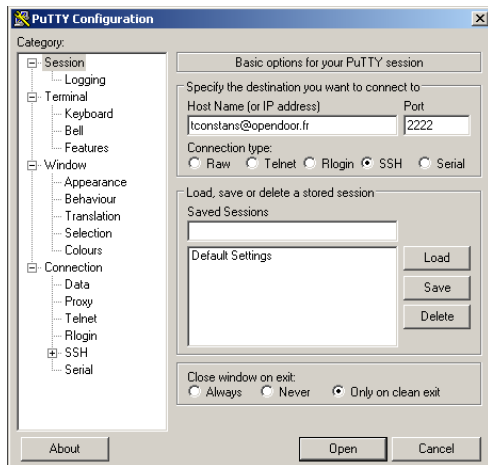
# Ssh depuis un poste windows



- ▶ *puttygen* permet de *générer* et d'*importer* des clés ssh.



# Ssh depuis un poste windows



- ▶ *putty* permet de sauvegarder, de gérer et de lancer des session ssh.
- ▶ Il supporte l'authentification par clé.

# Ssh depuis un poste windows

