

# Introduction aux systèmes GNU/Linux

## Séance 4

[inetdoc.net](http://inetdoc.net)



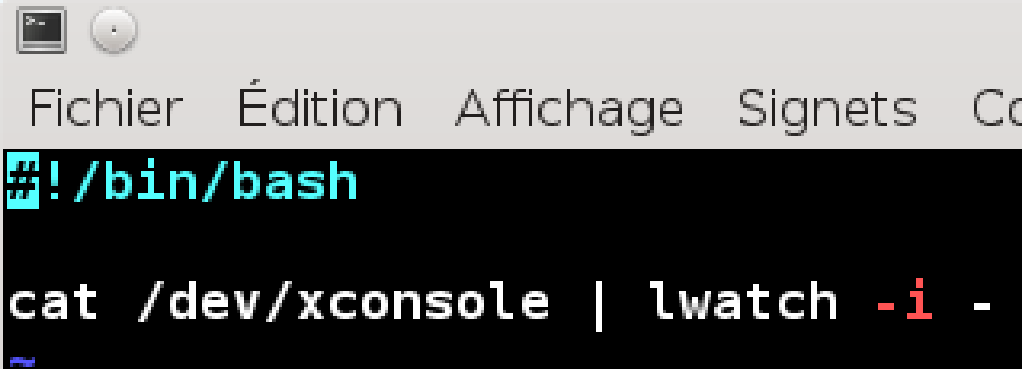
Philippe Latu / Université Toulouse 3 - Paul Sabatier  
Document sous licence GNU FDL v1.3  
<http://www.gnu.org/licenses/fdl.html>

# Plan séance 4

- Séance 4 - Configuration du Système GNU/Linux
  - Utiliser les ressources du shell Bash
  - Identifier & gérer les processus
  - Gérer les permissions sur les fichiers et répertoires
  - Compiler une application à partir de ses sources
- Manipulations réalisables sur machines virtuelles
  - Comparaison entre applications empaquetées avec configuration et applications compilées localement sans configuration

# Shell Bash

- GNU BASH → Bourne-again Shell
  - Interpréteur de commandes
    - IEEE POSIX P1003.2/ISO 9945.2 Shell and Tools standard
  - Environnement de développement
    - Fonctions et alias
    - Arithmétique et tableaux
    - Manipulations de chaînes de caractères
  - Documentation shell Bash
    - Manuel de référence
      - <http://www.gnu.org/software/bash/manual/>
    - Advanced Bash Scripting
      - <http://www.tldp.org/LDP/abs/html/>



```
Fichier  Édition  Affichage  Signets  Co
# !/bin/bash
cat /dev/xconsole | lwatch -i -
```

```
$ man sh
```

Documentatio  
n syntaxe

```
$ help
```

Documentation  
commandes internes

# Shell Bash

- Éditions et corrections en ligne de commande
  - Synthèse des «styles» Emacs, vi, IOS et autres shells
  - Touche tabulation → *auto completion*
  - Historique et rappel des commandes antérieures
    - Commande **history**
    - Séquences de touches
      - Ctrl+R, Ctrl+A, Ctrl+E
      - Shift+PageUp, Shift+PageDown, flèches haut et bas
- Séquencement entre les opérations
  - Tubes ou *pipes* → |
  - Enchaînements logiques → ; → && → ||
  - Commandes internes → jobs, suspend

# Shell Bash

- Exemple de script shell
  - Nettoyage des bases de données du navigateur Mozilla Firefox

```
#!/bin/bash
command=`which sqlite3`
if [ ! -x "$command" ]; then
    echo "ERREUR : le programme $command est introuvable"
    exit 1
fi
if [ ! -z `pidof firefox-bin` ]; then
    echo "ERREUR : le navigateur est en cours d'exécution"
    exit 1
fi
for base in ~/.mozilla/firefox/*.default/*.sqlite; do
    $command $base "VACUUM" && echo "Base $(basename $base .sqlite) nettoyée."
done
exit 0
```

Affectation d'une variable

Test sur la présence d'un programme exécutable

Test sur la présence d'un processus actif

Boucle de traitement des fichiers de bases de données

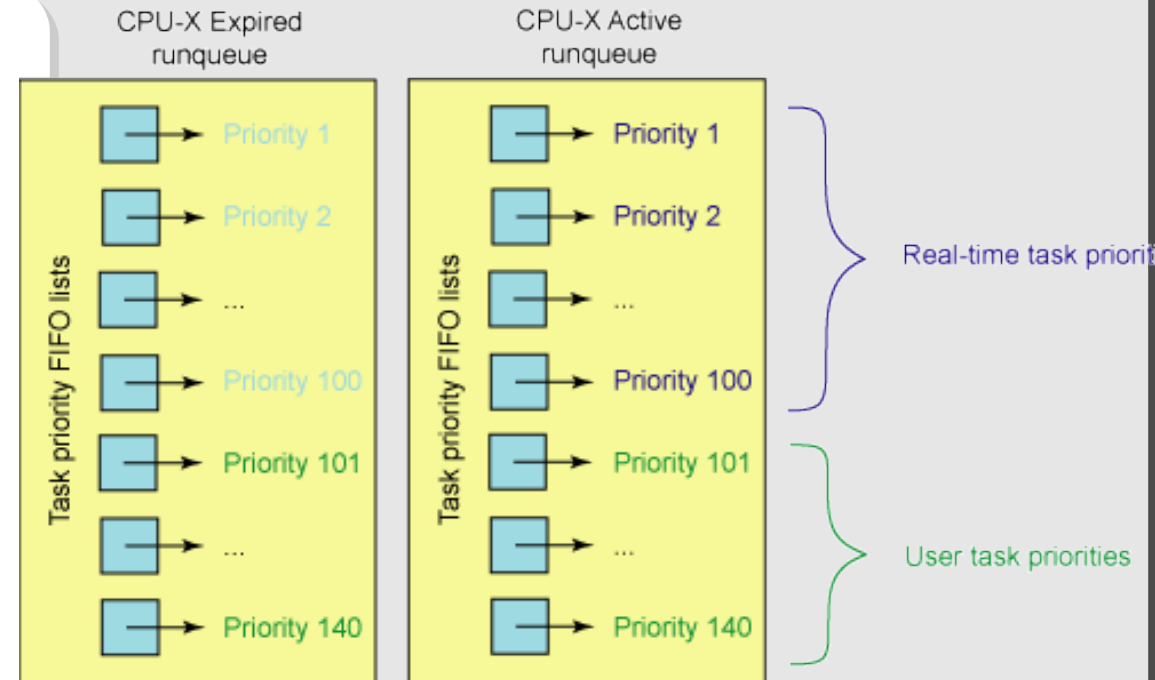
# Shell Bash

## ▪ Applications

- Comment lister l'historique des commandes ?
- Quel est l'effet de la séquence de touche Ctrl+A ?
- Comment rappeler une commande dans l'historique ?
- Comment modifier le fichier `~/.bashrc` pour afficher les fichiers et répertoires en couleur ?
- Quel est l'effet de la séquence de touche Ctrl+D ?
- Quelles sont les différences entre les commandes `set` et `env` ?
- Comment accéder à la documentation de chacune de ces commandes ?

# Processus

- Processus → programme en cours d'exécution
  - Fonction Unix de base → partage des ressources du système entre différents programmes
  - Ordonnanceur (*Scheduler*) → attribution des tranches de temps processeur en fonction du type de tâche
  - Fonctions multi-tâches préemptives du noyau Linux
    - Planification de l'exécution des processus
    - Contrôle au début et à la fin de chaque tranche de temps processeur



# Processus

- Gestion des processus

- Qui est le propriétaire d'un processus ?
- Quelles sont les ressources utilisées par un processus ?
- Comment changer le niveau de priorité d'un processus ?
- Comment tuer un processus défectueux ?

- Commandes de gestion des processus

- Qui, quoi, combien
  - ps, w, top, iotop, htop, surveillance système environnement graphique

- Priorités

- nice, renice, ionice

- Signalisation et arrêt

- kill, killall

- Utilisation mémoire

- free, cat /proc/meminfo, vmstat

```
1 [||| 1.3%] Tasks: 42, 86 thr; 1 running
2 [ 0.0%] Load average: 0.07 0.04 0.05
Mem [||||| 114/1002MB] Uptime: 01:04:41
Swp [ 0/3811MB]

PID USER PRI NI VIRT RES SHR S CPU% MEM% TIME+ Command
3143 root 20 0 24220 2504 1344 R 0.0 0.2 0:00.74 htop
1 root 20 0 10632 816 676 S 0.0 0.1 0:00.51 init [2]
299 root 20 0 21632 1608 772 S 0.0 0.2 0:00.09 udevd --daemon
```



# Processus

- Applications
  - Commande `ps`
    - Comment visualiser les processus, les propriétaires et les terminaux ?
    - Quelle est la signification de la commande `ps faux` ?
  - Commandes `kill` et `killall`
    - Quelle est la signification du terme signal ?
    - Comment relancer un processus ?
    - Comment tuer un processus «en force» ?
- Processus ou services inutiles
  - Comment supprimer un service inutile de façon permanente ?
  - Comment caractériser le gain en occupation mémoire correspondant ?



# Permissions sur le système de fichiers

- Masque des permissions de base → 10 indicateurs
  - Partant de la gauche
    - Premier indicateur → nature de l'objet
      - fichier, répertoire, périphérique ou socket Unix
    - Autres indicateurs → droits
      - lecture, écriture, exécution
      - Propriétaire, groupe et autre

d rwx rwx rwx

Permissions for others  
Permissions for group  
Permissions for owner

- → file | d → directory | l → symlink |  
b → block | c → character | s → socket

# Permissions sur le système de fichiers

- Édition du masque des permissions
  - Commandes usuelles
    - `ls` → visualisation
    - `chown`, `chgrp` → changement de propriétaire ou de groupe
    - `chmod` → changement de masque
    - `umask` → masque utilisateur utilisé par défaut lors de la création d'objets
  - Codage des permissions
    - Notation littérale
      - `r` → read → droit de lecture
      - `w` → write → droit d'écriture
      - `x` → execute → droit d'exécution
    - Notation en octal
      - `r` →  $2^2$  → 4
      - `w` →  $2^1$  → 2
      - `x` →  $2^0$  → 1

```
$ touch emptyfile
$ ls -l emptyfile
-rw- r-- r-- 1 etu etu 0 mai 16 11:20 emptyfile
420 400 400
6 4 4 → 644
$ chmod +x emptyfile
-rwx r-x r-x 1 etu etu 0 mai 16 11:20 emptyfile
421 401 401
7 5 5 → 755 → chmod +x = chmod 755
```

# Permissions sur le système de fichiers

- Applications
  - Commande `ls`, donner un exemple de
    - programme exécutable
    - lien symbolique
    - périphérique en mode caractère
    - périphérique en mode bloc
    - socket Unix
  - Permissions sur les fichiers et répertoires
    - Donner les valeurs numériques des masques d'un fichier de données et d'un programme
    - Quel est l'effet de l'instruction `umask 027` ?
  - Shell script Bash «Hello, World!»
    - Créer le fichier script `hello.sh` et le rendre exécutable

```
#!/bin/bash  
  
echo "Hello, World!"
```

# Permissions sur le système de fichiers

- Masque étendu
  - 3 bits supplémentaires → extension des permissions
    - **SUID** : Set User ID bit
    - **SGID** : Set Group ID bit
    - directory **Sticky bit**
  - Ces 3 bits prennent la place du bit d'exécution **x**
  - Pour le propriétaire du fichier
    - **s** indique qu'il a aussi le droit d'exécution
    - **S** indique qu'il n'a pas le droit d'exécution
  - Pour le groupe du fichier
    - **s** indique qu'il a aussi le droit d'exécution
    - **S** indique qu'il n'a pas le droit d'exécution
  - Directory Sticky bit
    - Utile pour les répertoires partagés
    - Un utilisateur ne peut effacer que les fichiers qu'il a créé

# Permissions sur le système de fichiers

## ▪ Applications

- Quel est le rôle du masque étendu pour les objets suivants ?  
/usr/bin/passwd, /usr/bin/wall et /tmp
- Comment activer le bit SUID sur un fichier ?
  - Créer un fichier test avec la commande `touch`.
  - Donner les options de la commande `chmod`
- Comment activer le bit SGID sur un répertoire ?
  - Créer un répertoire avec la commande `mkdir`
  - Donner les options de la commande `chmod`
- Comment activer le Directory Sticky bit sur un répertoire ?
  - Créer un répertoire avec la commande `mkdir`
  - Donner les options de la commande `chmod`

# Compilation d'une application

- Compilation d'un logiciel libre
  - 3 étapes classiques → ./configure ; make ; make install
  - Téléchargement & décompression des sources
    - Commande **tar** → *tape archive*
    - Commandes gzip ou bzip2 → outils de décompression
  - Étape ./configure
    - Garantit la portabilité du logiciel
    - Détermine les caractéristiques du système cible
      - Chaîne de développement, architectures, outils
    - Vérifie la présence des bibliothèques nécessaires
    - Génère les **Makefiles**



# Compilation d'une application

- Étape make
  - Compilation des modules du logiciel
  - Correspondance entre bibliothèques et programmes
- Étape make install
  - Copie des fichiers de l'arborescence de développement vers l'arborescence système
  - Exécutables, bibliothèques, pages de manuels
- Distinction entre `/usr` et `/usr/local`
  - `/usr` → fichiers et répertoires fournis par les paquets
  - `/usr/local` → fichiers et répertoires compilés localement
  - `/usr/local/src` → fichiers et répertoires de l'arborescence de développement
  - `/usr/local/bin` → figure dans la variable `$PATH` de l'utilisateur

# Compilation d'une application

- Exemple : analyseur réseau wireshark
  - Télécharger les sources depuis <http://www.wireshark.org>
  - Créer l'arborescence de développement
    - Décompresser l'archive dans le répertoire `/usr/local/src`
      - Quels sont les droits nécessaires à l'utilisation du répertoire `/usr/local/src` ?
      - Quelles sont les options de la commande `tar` à utiliser ?
  - Consulter les fichiers d'aide à l'installation
    - README, README.linux, INSTALL, INSTALL.configure
  - Identifier les bibliothèques nécessaires à la compilation
    - Utiliser les informations données par la commande `./configure --help`

# Compilation d'une application

- Recherche et installation des paquets de bibliothèques
  - Identifier le rôle des bibliothèques gtk et pcap
  - Installer les paquets de développement correspondants
    - Rechercher les paquets ayant pour nom
      - `libgtk.*dev`
      - `libpcap.*dev`
    - Installer les deux paquets et reprendre l'étape configure
    - Visualiser le bilan de l'étape configure
- Compilation des sources avec la commande `make`
  - Rechercher l'option `-j` dans les pages de manuels

# Compilation d'une application

- Test de l'application dans l'arborescence de développement
  - Exécuter wireshark depuis le répertoire courant
    - Contrôler les droits d'utilisation de l'écran le niveau utilisateur
- Installation avec la commande `make install`
  - Installation dans l'arborescence `/usr/local`
  - Repérer les chemins des bibliothèques
  - Repérer les chemins des pages de manuel
  - Reprendre le test d'exécution de l'application
  - Exécuter la commande `'ldconfig'`
    - Parcours de l'arborescence des bibliothèques locales

# Application graphique & droits

- Utilisation de l'analyseur réseau

- Contexte métier

- Capture en mode console
      - Droits étendus
      - Accès direct aux interfaces réseau
    - Analyse en mode graphique
      - Droits utilisateur normaux
      - Accès au fichier au fichier de capture

- Contexte travaux pratiques

- Capture & analyse en mode graphique
      - Droits étendus à partir de l'application graphique
      - Exécution du programme wireshark au niveau super-utilisateur
      - Délégation de l'accès à l'écran de l'utilisateur normal

L'écran est réservé à l'utilisateur connecté

```
$ xhost  
access control enabled, only authorized  
clients can connect  
$ xhost +local:  
non-network local connections being added to  
access control list  
$ xhost  
access control enabled, only authorized  
clients can connect  
LOCAL:
```

Les utilisateurs locaux non-réseau ont accès à l'écran

# Bilan séance 4

- Shell Bash
  - Interpréteur de commandes aux fonctions étendues
  - Interface de base de l'administration système → contexte infrastructure
- Processus
  - Tout programme en cours d'exécution a l'«identité» de son propriétaire
- Droits sur le système de fichiers
  - Principes de gestion des droits Unix
  - Compromis efficacité/simplicité
- Compilation d'une application à partir de ses sources
  - À éviter ! Les paquets sont là pour nous aider
  - Connaissances des dépendances et des outils de développement

# Ressources

- Beaucoup mieux que la compilation à partir des sources
  - Debian Packaging Tutorial
    - Comment construire un paquet
      - <http://www.debian.org/doc/manuals/packaging-tutorial/packaging-tutorial.en.pdf>
  - Debian Handbook – chapter 15
    - Creating a Debian Package
      - <http://static.debian-handbook.info/browse/stable/debian-packaging.html>

