

# [inetdoc.LINUX]

<http://www.linux-france.org/prj/inetdoc>

## Exploration GNU/Linux - Séance 4

Shell Bash

Processus et Permissions sur le système de fichiers

Compilation d'une application à partir des sources



Philippe Latu

[philippe.latu@linux-france.org](mailto:philippe.latu@linux-france.org)

IUT 'A' Paul Sabatier - STRI

# Administration du système GNU/Linux

- Objectifs

- Utiliser les ressources du shell BASH
- Visualiser et gérer les processus
- Visualiser et gérer les permissions sur les fichiers
- Compiler une application à partir des sources

# Ressources du shell BASH

- BASH : Bourne Again Shell
- Shell BASH = interpréteur de commandes
  - IEEE POSIX P1003.2/ISO 9945.2 Shell and Tools standard
- Shell BASH = environnement de développement
  - Fonctions et alias
  - Arithmétique et tableaux
  - Manipulations de chaînes de caractères
- Documentation shell BASH
  - Manuel de référence
    - ▶ <http://cnswww.cns.cwru.edu/~chet/bash/bashtop.html>
  - Advanced Bash-Scripting Guide
    - ▶ <http://tldp.org/LDP/abs/html/>

\$ man sh      # Complet sur la syntaxe

\$ help        # Commandes internes au Shell

# Ressources du shell BASH

- Édition et correction des lignes de commandes
  - Synthèse des «styles» Emacs, vi, IOS et autres shells
  - Tabulation = appel fonctions 'auto\_completion'
- Historique et rappel des commandes
  - Commande history
  - Séquences de touches
    - ▶ Ctrl+R, Ctrl+A, Ctrl+E
    - ▶ Shift+PageUp, Shift+PageDown, flèches haut et bas
- Contrôle des tâches
  - Tubes ou pipes = '|'
  - Enchaînements = ';', '&&', '||'
  - Commandes internes = jobs, suspend

# Ressources du shell BASH

- Exemple de script en shell BASH
  - Ancien script système /etc/init.d/modutils

```
PATH="/sbin:/bin:/usr/sbin:/usr/bin"
```

```
[ -f /proc/modules ] || exit 0
```

```
[ -e /sbin/depmod ] || exit 0
```

```
echo -n "Calculating module dependencies... "
```

```
depmod -a > /dev/null
```

```
echo "done."
```

```
# Loop over every line in /etc/modules.
```

```
echo -n 'Loading modules: '
```

```
(cat /etc/modules; echo) | # make sure there is a LF at the end
```

```
while read module args
```

```
do
```

```
    case "$module" in
```

```
        \#*|"") continue ;;
```

```
    esac
```

```
    echo -n "$module "
```

```
    modprobe $module $args
```

```
done
```

```
echo
```

# Ressources du shell BASH

- Environnement utilisateur - BASH

- Applications

- ▶ Lister l'historique des commandes ?
- ▶ Quel est l'effet de la séquence Ctrl+R ?
- ▶ Éditer le fichier ~/.bashrc pour obtenir les listes de fichiers en couleur
- ▶ Tester la touche Tab sur un chemin et/ou un nom de fichier
- ▶ Quel est l'effet des séquences de touches Ctrl+A, Ctrl+E et Ctrl+D ?
- ▶ Quelles sont les différences entre les commandes set et env ?
- ▶ Comment accéder à la documentation de ces 2 commandes ?

# Processus

- Définitions

- Processus = programme en cours d'exécution
- Ressources systèmes partagées entre processus
- L'ordonnanceur attribue les tranches de temps processeur
- Fonctions multi-tâches préemptives du noyau Linux
  - ▶ Planification de l'exécution des processus
  - ▶ Contrôle en début et fin de chaque tranche de temps processeur
- Contrôler le fonctionnement multi-tâche et la stabilité
  - ▶ Qualité du système d'exploitation

# Processus

- Tous les programmes ne sont pas parfaits
  - Gérer les processus.
    - ▶ Qui sont les propriétaires des processus ?
    - ▶ Quelles sont les ressources utilisées ?
    - ▶ Comment changer un niveau de priorité ?
    - ▶ Comment tuer un processus défectueux ?
- Commandes de gestion des processus
  - Qui, quoi, combien
    - ▶ w, ps, top, htop, Surveillance système KDE
  - Priorités
    - ▶ nice, renice
  - Signalisation et arrêt
    - ▶ kill, killall
  - utilisation mémoire
    - ▶ free, cat /proc/meminfo

# Processus

- Applications à l'aide des pages de manuel
  - Commande ps
    - ▶ Visualiser les processus, les propriétaires et les terminaux
    - ▶ Quelle est la signification de la commande 'ps xaf' ?
  - Commandes kill et killall
    - ▶ Quelle est la signification du terme signal ?
    - ▶ Comment relancer un processus ?
    - ▶ Comment tuer un processus «en force» ?
  - Processus/services inutiles
    - ▶ Relever un processus inutile en cours d'exécution
    - ▶ Retirer le paquet correspondant

# Permissions sur le système de fichiers

- Un objet = «masque» de permissions

- commande ls -lA

\$ ls -lA

```
drwx----- 2 etu etu 4096 Jan 1 00:04 mail
drwxrwxr-x 3 etu etu 4096 Nov 18 09:54 public_html
-rw-rw---- 1 etu etu 136430 Feb 6 16:52 trash.file
```

```
  ^      ^  ^  ^      ^      ^
```

```
 |      |  |  |      |      |
```

permissions owner group size date & time filename

- Chaque utilisateur appartient à plusieurs groupes

- ▶ commande 'id'

\$ id

```
uid=1000(etu) gid=1000(etu) groups=1000(etu),29(audio)
```

```
  ^      ^      ^
```

```
 \__ private _/      public group(s)_/
```

# Permissions sur le système de fichiers

- Masque des permissions de base = 10 indicateurs
  - Premier indicateur = nature de l'objet
    - ▶ Fichier, répertoire, périphérique ou socket
  - Autres indicateurs = droits
    - ▶ Lecture, écriture et exécution
    - ▶ Propriétaire, groupe et autres (world)

d rwx rwx rwx

^ ^ ^ ^

| | | '-----> Permissions for world

| | '-----> Permissions for group

| '-----> Permissions for user

'-----> - = file, d = directory, \  
b = block, c = character, \  
l = symlink, s = socket

# Permissions sur le système de fichiers

- Commandes de manipulations
  - ls, chmod, chown et umask
- Codage des permissions
  - Chiffres : 1 = exec, 2 = write, 4 = read
  - Lettres : x = exec, w = write, r = read

```
$ touch toto
```

```
$ ls -l toto
```

```
-rw-r--r--  1 etu  etu      0 jun 11 20:20 toto
```

```
$ chmod +x toto
```

```
$ ls -l toto
```

```
-rwxr-xr-x  1 etu  etu      0 jun 11 20:20 toto
```

```
^^^^ ^^ ^
```

```
4214 14 1
```

```
-----
```

```
7 5 5
```

- chmod +x toto est identique à chmod 755 toto

# Permissions sur le système de fichiers

- Applications

- Commande ls

- ▶ Donner un exemple de programme exécutable
- ▶ Donner un exemple de lien symbolique
- ▶ Donner un exemple de périphérique
- ▶ Donner un exemple de fichier caché

- Masques de permissions

- ▶ Donner la valeur numérique du masque d'un fichier de données
- ▶ Donner la valeur numérique du masque d'un fichier exécutable

- Créer et rendre le script suivant exécutable

```
#!/bin/sh
```

```
echo "Hello World !"
```

# Permissions sur le système de fichiers

- Masque étendu

- 3 bits supplémentaires qui étendent les permissions
  - ▶ SUID : Set User ID bit
  - ▶ SGID : Set Group ID bit
  - ▶ directory Sticky bit
- Ces 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 ?
  - ▶ `/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 'testdir' avec la commande 'mkdir'
  - ▶ Donner les options de la commande `chmod`
- Comment activer le Directory Sticky bit sur un répertoire ?
  - ▶ Créer un répertoire 'testdir' avec la commande 'mkdir'
  - ▶ Donner les options de la commande `chmod`

# Permissions sur le système de fichiers

- Attention ! Masque étendu = danger
  - Bit SUID = gros problèmes de sécurité sur les services
    - ▶ Attaques de type «buffer overflow»
    - ▶ L'utilisation de ce bit est très encadrée
  - Utilisation d'arborescences dédiées par services
    - ▶ Exécution en «prison» ou chroot
    - ▶ <http://www.debian.org/doc/manuals/securing-debian-howto/>

# Compilation d'une application

- Compilation de logiciel libre
  - Étapes classique
    - ▶ `./configure ; make ; make install`
- Téléchargement et décompression des sources
  - Commande tar
  - Outils de (dé)compression (gzip|bzip2)
- Étape configure
  - Garantit la portabilité du logiciel
  - Détermine les caractéristiques du système
    - ▶ Compilateur, processeur, outils
  - Vérifie la présence des bibliothèques nécessaires
  - Génère les Makefiles

# Compilation d'une application

- Étape make

- Compilation des modules du logiciel

- ▶ Liens entre bibliothèques et modules

- Étape make install

- Copie des fichiers dans l'arborescence

- ▶ Exécutables
- ▶ Bibliothèques
- ▶ Pages de manuel

- Arborescence `/usr/local`

- Distinction entre

- ▶ Applications distribuées via les paquets
- ▶ Applications compilées localement

- Répertoire `/usr/local/bin`

- Doit figurer dans la variable `$PATH` de l'utilisateur

# Compilation d'une application

- Outils GNU 'autoconf', 'automake' et 'libtool'
  - Automatisation des tâches
  - Génération d'arborescences sources
  - Contrôles des dépendances sur les sources
    - ▶ <http://sources.redhat.com/autobook/>
- Diffusion d'applications portables
  - Entre architectures
  - Entre distributions
  - Entre branches Unix

# Compilation d'une application

- Exemple avec l'analyseur réseau WireShark
  - Retirer les paquets installés avec le nom wireshark\*
  - Télécharger les sources
    - ▶ <http://www.wireshark.org>
  - Déplacer le fichier des sources
    - ▶ Se connecter en mode super utilisateur
    - ▶ Répertoire /usr/local/src
  - Décompresser les sources
    - ▶ Commande 'tar'
    - ▶ Rechercher les options dans les pages de manuel
  - Consulter les fichiers d'aide à l'installation
    - ▶ README, README.linux, INSTALL, INSTALL.configure
  - Chercher les options particulières de l'étape configure
    - ▶ ./configure --help
  - Repérer les bibliothèques dont dépend l'application

# Compilation d'une application

- Identifier le rôle des bibliothèques
  - libgtk & libpcap
- Retrouver les paquets des bibliothèques
  - Distinction entre
    - ▶ Bibliothèque d'exécution
    - ▶ Bibliothèque de développement

```
# dpkg -l "libgtk*dev"
```

```
# dpkg -l "libpcap*"
```

- Installer les paquets correspondant

```
# apt-get install libgtk2.0-dev
```

```
# apt-get install libpcap-dev
```

# Compilation d'une application

- Exécuter la commande configure

# ./configure

- Visualiser le bilan de l'exécution
  - Contrôler les problèmes de dépendances
  - Vérifier la liste des options retenues
- Reprendre l'exécution de configure
  - Tant qu'il y a des messages d'erreur
  - Si des options doivent être ajoutées

# Compilation d'une application

- Exécuter la commande make
- Tester l'application
  - Exécuter wireshark depuis le répertoire courant

# ./wireshark

- !! Attention aux droits sur le partage de l'écran
- Exécuter 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

- **Activité réseau = accès direct aux interfaces**
  - Travail en mode super-utilisateur
  - Recours au bit SUID proscrit
  - Partage de l'écran entre utilisateurs
    - ▶ Commande 'xhost'

\$ 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:

- Partage de l'écran autorisé
  - ▶ Utilisateurs locaux non réseau
- Tester à nouveau wireshark

# Synthèse

- Shell BASH
  - ▶ Interpréteur de commandes aux fonctions très étendues
- Processus
  - ▶ Droits limités pour les propriétaires des processus
- Permissions sur le système de fichiers
  - ▶ Principe de gestion des droits UNIX
  - ▶ Compromis simplicité/efficacité
- Compilation d'application à partir des sources
  - ▶ Bonne connaissance des dépendances entre bibliothèques et application
  - ▶ Bonne connaissance de l'environnement de développement
- Pour débiter, utilisez les paquets !