Formation Linux LPIC-1
Compilation

Guillaume Allègre
Guillaume.Allegre@silecs.info

Grenoble INP - Formation Continue

2014

Licence Ceative Commons By - SA

- Vous êtes libre de
  - **partager** — reproduire, distribuer et communiquer l'oeuvre
  - **remixer** — adapter l'oeuvre
  - d'utiliser cette œuvre à des fins commerciales
- Selon les conditions suivantes
  - **Attribution** — Vous devez attribuer l'oeuvre de la manière indiquée par l'auteur de l'oeuvre ou le titulaire des droits (mais pas d'une manière qui suggérerait qu'ils vous soutiennent ou approuvent votre utilisation de l'oeuvre).
  - **Partage à l'identique** — Si vous modifiez, transformez ou adaptez cette oeuvre, vous n'avez le droit de distribuer votre création que sous une licence identique ou similaire à celle-ci.

`http://creativecommons.org/licenses/by-sa/3.0/deed.fr`

Contribuer - Réutiliser

Ce document est rédigé en LaTeX+ Beamer.

Vous êtes encouragés à réutiliser, reproduire et modifier ce document, sous les conditions de la licence *Creative Commons, Attribution, Share alike 3.0* précédemment décrite.

J'accepte volontiers les remarques, corrections et contributions à ce document

Vous pouvez obtenir les sources LaTeXde ce document sur le dépôt Mercurial :
http://hg.silecs.info/hg/public/formations/linux/
où vous pouvez naviguer ou télécharger une archive.
Une version PDF est disponible sur
http://www.silecs.info/dld/lpi/

Révision 110 :cf76d07e0f73

Linux Professional Institute Certification

Ce document est un support de formation adapté à la préparation de la certification LPIC-1 (101 et 102).
Ce document **n'est pas** un support agréé officiellement par le LPI.

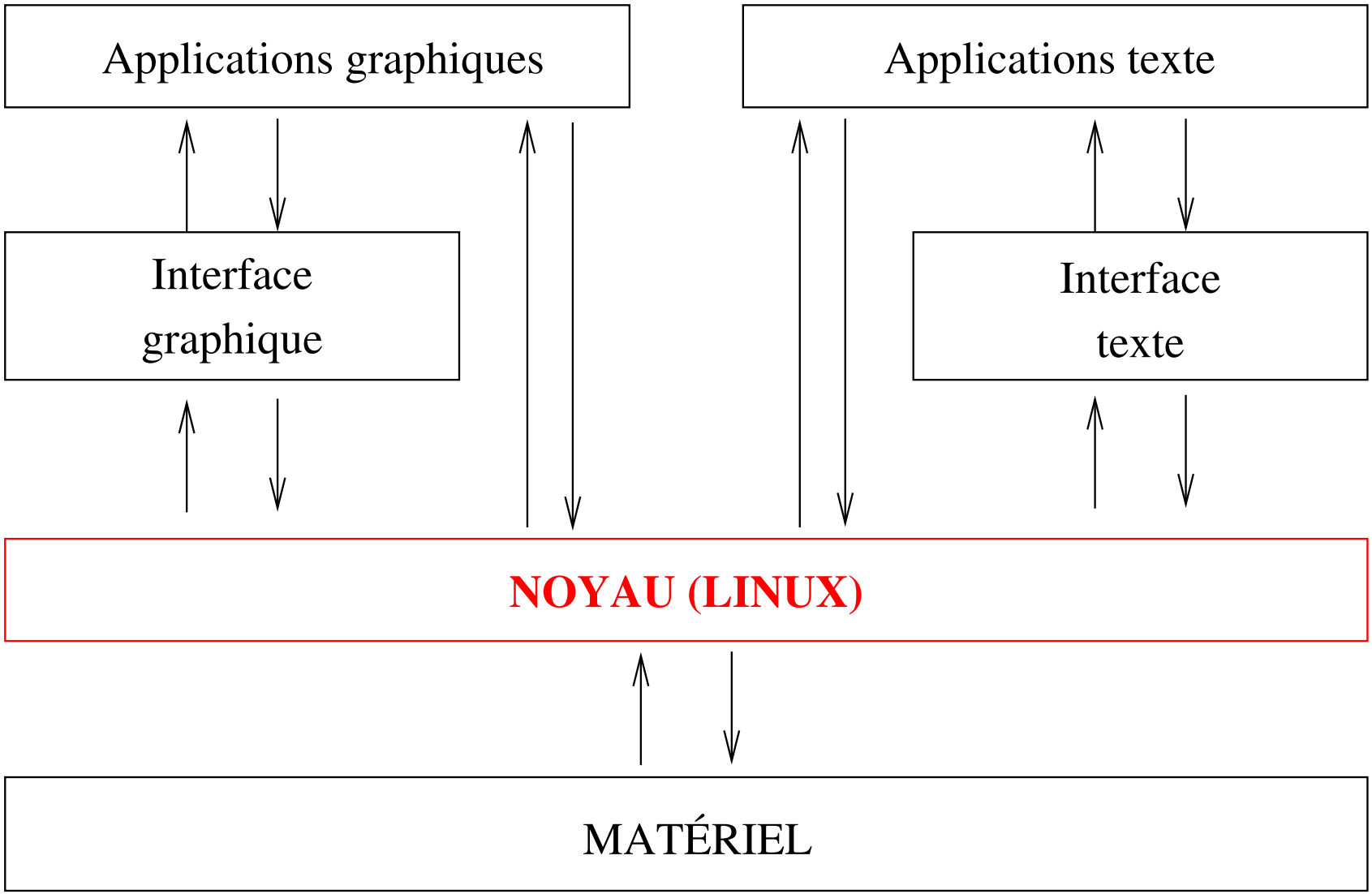
Chaque transparent directement lié au programme LPI porte la référence de l'item LPI correspondant (par exemple **105.3**) sur la ligne de titre. Les documents de référence sont *Objectifs détaillés des examens* LPIC 101 et LPIC 102, révision d'avril 2009 (traduits en français) :
`http://www.lpi-francophonie.org/spip.php?rubrique19`

L'ordre des notions abordées diffère de celui du programme LPI. Le parti-pris de ce document est de se concentrer d'abord sur la maîtrise des outils en ligne de commande (utilisateurs), puis seulement sur les outils d'administration.

L'auteur (Guillaume Allègre) est certifié LPIC-1.

Qu'est-ce que Linux ?

Architecture d'un système d'exploitation



Une histoire de famille : Unix

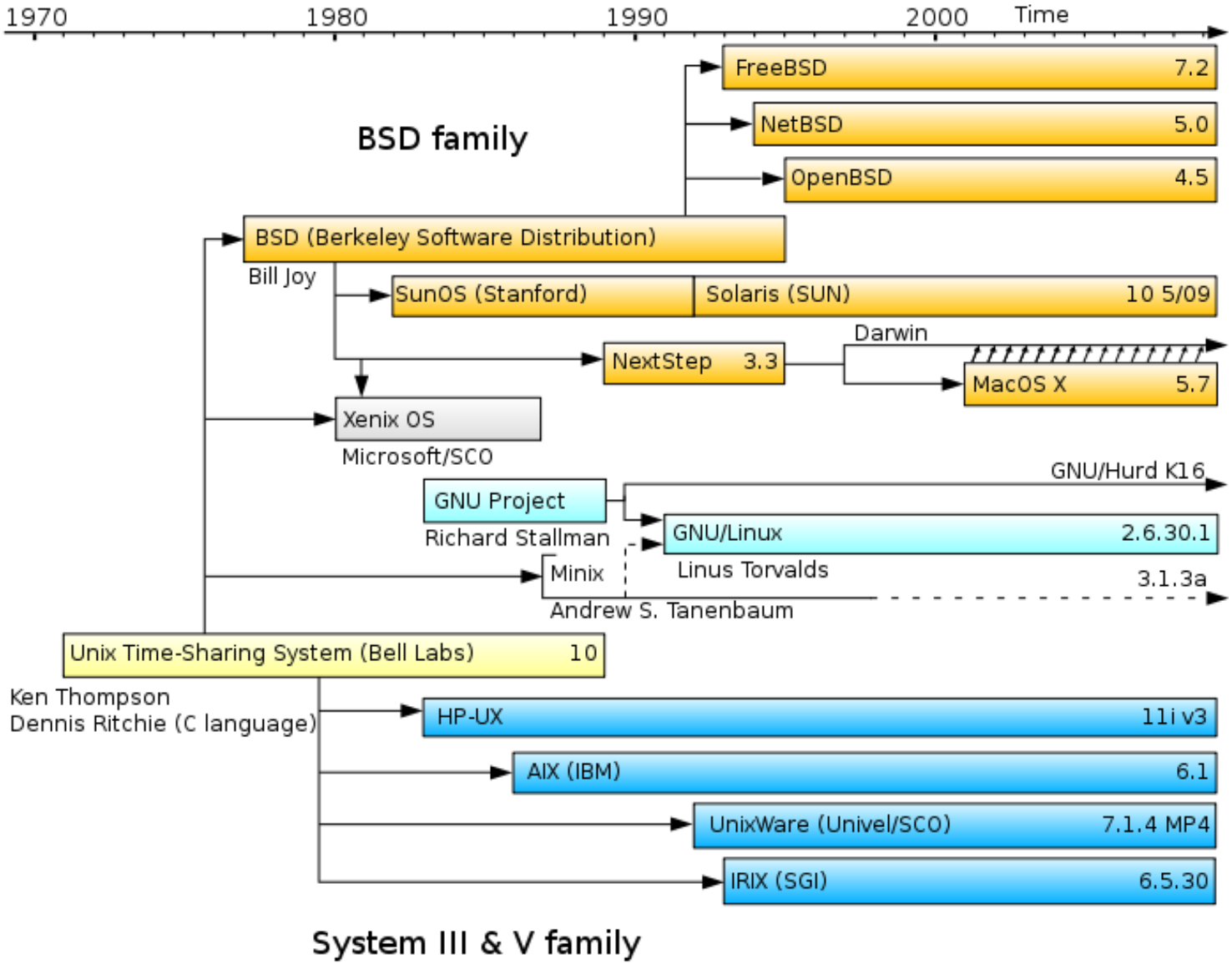
UNIX en quelques points :

1. apparu en 1969 à AT&T - Bell Labs., K. Thompson, D. Ritchie
2. beaucoup de dérivés : Solaris, AIX, BSD, OS X...
3. conçu comme un système professionnel :
   - orienté réseau,
   - multi-tâches,
   - multi-utilisateurs.
4. trois survivants propriétaires : Solaris (Sun), AIX (IBM), HP-UX

Une normalisation : POSIX (IEEE 1003) 1985-1998

1. 17 thèmes : Core, Real-time, Threads, Shell...
2. évolutions : POSIX :2001, POSIX :2004, POSIX :2008

Une brève histoire d'Unix

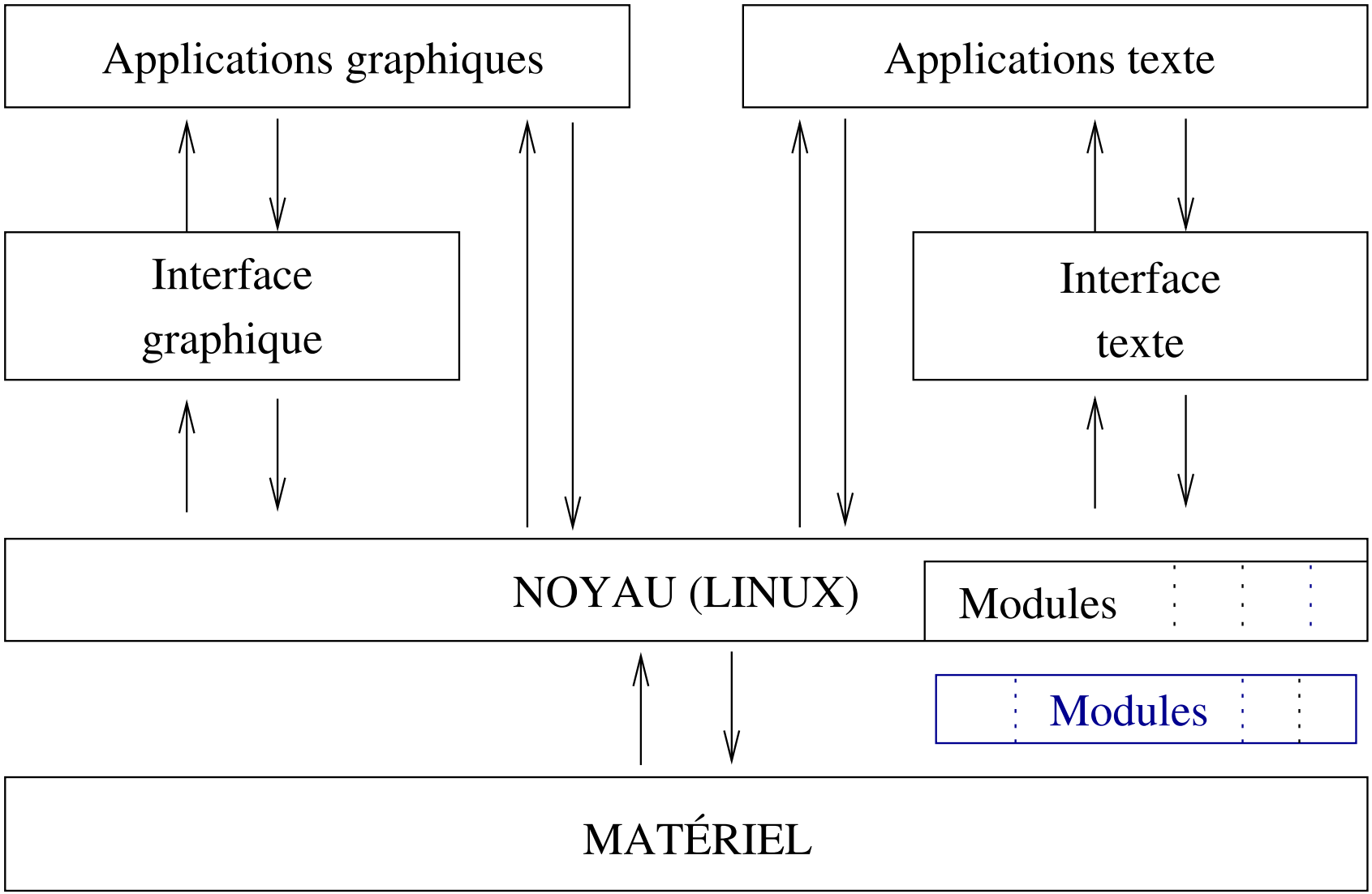


Domaine Public - Wikimedia Commons - Unix history.en.svg

Les spécificités de Linux

- créé en 1991 par *Linus Torvalds*, étudiant finlandais.
- logiciel libre
  - inscrit dans la mouvance GNU
  - sous licence GPL depuis 1992
  - fer de lance du logiciel libre
- développement décentralisé et collaboratif
- modulaire : chargement d'extension du noyau à la demande (pilotes...)
- portable : compatible avec un très grand nombre d'architectures.

Le système Linux



Principales différences GNU/Linux / Windows

1. Un ensemble très modulaire vs. un bloc monolithique
2. Une seule arborescence (*tout est fichier*)
3. Fichiers de configuration et éditeurs de texte
   (pas de base de registres)
4. Importance de la ligne de commande (*une tâche, un outil*)
5. Profondément **réseau** et **multi-utilisateurs**

Linux et le libre

- Linux est un système d'exploitation sous **licence libre**
  1. liberté d'usage, sans restriction
  2. liberté d'étude du logiciel et de modification
  3. liberté de copie et diffusion
  4. liberté de diffusion des modifications

- Pour 2. : importance du **code source**
- Sphère privée (1-2) / sphère publique (3-4)

- Licence **GPLv2** : General Public License
  Il existe d'autres licences libres (ex : BSD, MPL...)

- Projet GNU : Le complément du noyau...

Le projet GNU : *GNU's Not Unix*

- Origine (1983) : réimplémentation libre des utilitaires Unix
  - `glibc` + `gcc` : GNU C library + GNU C Compiler
  - `binutils` (ld, as, gprof, nm, ar, strings...), make, gdb...
  - `coreutils` (ls, chmod, sort, du, nice...), grep, sed, awk
  - `bash` : shell compatible sh

- Récemment : focalisation sur les projets "stratégiques"
  - GNU Hurd : noyau libre (pas opérationnel, cf. Linux)
  - Gnu Privacy Guard : crypto personnelle (alternative à PGP)
  - Gnome : environnement de bureau (alternative à KDE)
  - Gnash : lecteur Flash libre (alternative à Adobe...)
  - ...

- Logiciels indépendants
  - Emacs (1976-) : éditeur texte original, alternatif à `vi`
  - GIMP : retouche d'images
  - Dia : conception de diagrammes
  - ...

Linux et le libre

- Il existe des logiciels propriétaires pour Linux (ex. serveur Oracle)
- Il existe des logiciels libres pour Windows... (ex. Apache, Mozilla Firefox, OpenOffice.org)
- Il existe d'autres OS libres (ex. FreeBSD)
- Libre n'est pas gratuit
  - parfois si : Linux est libre **et** gratuit
  - *freeware* : gratuit, pas libre (code source)
  - développements à façon : libre, pas gratuit...

Les distributions Linux

Leur rôle :

- Simplifier la vie de l'administrateur.

Une distribution comprend :

- le noyau Linux
- un système d'installation
- des logiciels applicatifs
- des outils d'administration

- Éventuellement
  - un support physique (boîte, CDROM, documentation...)
  - des services (maintenance, hotline, formation...)

Les distributions Linux - Diversité (2)

Près de 400 distributions actives.

Cf. `http://distrowatch.com/` et `http://futurist.se/gldt/`

Causes de diversité :

1. Modèle de développement
   - communautaire : Slackware, Debian et certaines dérivées...
   - commerciale : la plupart des autres
2. Modèle d'administration
   - Installation des logiciels (.deb / .rpm / .tar.gz)
   - Services (Redhat / Fedora)
3. Spécialisation
   - Autonome : Knoppix, Kaella
   - Grand public : Ubuntu
   - Sécurité réseau : IP Cop
   - Localisation : Mandriva
   - Dépannage : System Rescue
   - Recompilation (performances) : Gentoo

Les distributions : la famille RedHat

- RedHat Linux (ancien modèle) : RH 1.0 (1994) à RH 9 (2003)
  - mise au point du format RPM (RedHat Package Manager)
- RedHat Enterprise Linux (RHEL) : depuis RHEL 3 (2003)
  - dernière : RHEL 6.2 (déc. 2011)
  - plusieurs variantes : Desktop, Workstation, ES, AS...
- Fedora (Core)
  - version communauté
  - dével. rapide (env. 2/an) depuis FC 1 (nov. 2003)
  - dernière : Fedora 16 (nov. 2011)
- CentOS
  - clone de RHEL, sans le service
  - utilise les sources fournies par RedHat
- autres utilisatrices de RPM : Mandriva, Novell SuSE...

Les distributions : la famille Debian

- Debian GNU/Linux : 1.0 (1996) à 6.0 Squeeze (fév. 2011)
  - collaborative et non commerciale
  - essentiellement libre
  - format de paquets (avancé) .deb
  - dépôts et installation réseau
  - mises à jour régulières (6.0.4 jan. 2012)

- Ubuntu : commerciale (Canonical LTD, GBM)
  - installation simplifiée
  - deux sorties par an (ex. 11.04 et 11.10)
  - partiellement compatible Debian
  - basée sur Gnome, choix restreint de paquets

- Knoppix : distribution autonome (*live*)
  - s'exécute sans installation (depuis le CD et la RAM)
  - peut s'installer et se transformer en Debian

Administration Linux : les paquets

Chaque distribution propose un système d'installation de logiciels via des paquets (.deb / .rpm / .tar.gz).

Avantages :

- Normalisation
- Simplification
- Gestion des dépendances
- Mise à jour centralisée

Remarque : possible d'installer un programme sans ce procédé.

Un effort de normalisation pour Linux

- Linux Standard Base (LSB)
  - 2001 (1.0) - 2011 (4.1) ...
  - dérivée / inspirée de POSIX
  - indépendante des distributions (mais RPM-centrée)
  - normalisation des composants (bibliothèques...)
  - normalisation de la hiérarchie (FHS)
  - fourniture de tests de compatibilité
- Linux Foundation
  - créée en 2007 : fusion de l'OSDL et du FSG
  - sponsorise Linus Torvalds et d'autres développeurs
  - édite la LSB et d'autres documents de référence (OpenPrinting...)

Les communautés du libre...

- Notion de communauté
  - modèle propriétaire : césure développeurs / utilisateurs
  - modèle libre : tous les intermédiaires
- Participation à la communauté
  - le « pot commun » : mutualisation et réciprocité
  - support informel (forums, listes de diffusion)
  - rapports de bugs (et plus)
- Émergence d'outils techniques
  - Internet et communication (mail, newsgroups)
  - Gestionnaires de versions (code source)
  - Suivi de bugs / de tickets (Bugzilla...)
  - SourceForge, GForge...

Logiciel libre : économie de services

- Économie de l'immatériel
  - Une idée n'est pas un bien matériel
  - Le partage n'appauvrit pas
  - Le logiciel "en boîte" est un leurre
- Des modèles économiques multiples
  - Constructeur : vend du matériel, donne le logiciel
  - Services : expertise, formation, développements sur mesure
  - Éditeur
    - hébergement (Software as a Service), *cloud*
    - audit, expertise
    - double licence, licence "chronodégradable"
- Quelques points délicats
  - Relations éditeur / communauté
  - Conditions de contribution
  - L'*open source* comme argument marketing

La "professionnalisation" de Linux

- Linux Foundation
- Linux Standard Base
- Linux Professional Institute : certification
  - Linux Essentials (2012)
  - LPIC-1 : administrateur junior
  - LPIC-2 : administrateur avancé
  - LPIC-3 : administrateur senior (3 spécialisations...)

Avantages du libre

- Éthique : collaboration, partage
  concerne : enseignement, administrations...
- Économie : redéploiement coûts achat vers services (formation, support)
- Pérennité et indépendance : moins lié à un éditeur
- Souplesse : adaptabilité aux besoins
- Mutualisation (coûts de développement)
  concerne : administration, collectivités locales...

Linux au démarrage *101.2*

En général (poste de travail) :

1. BIOS / EFI...
2. Chargeur de démarrage (GRUB ou LILO)
3. Mode texte
4. Mode graphique
5. Authentification par login + mot de passe
6. Bureau utilisateur (KDE, Gnome, XFCE...)

Linux au démarrage *101.2*

En général (poste de travail) :

1. BIOS / EFI...
2. Chargeur de démarrage (GRUB ou LILO)
3. Mode texte
4. Mode graphique
5. Authentification par login + mot de passe
6. Bureau utilisateur (KDE, Gnome, XFCE...)

On peut aussi avoir (serveur) :

1. BIOS / EFI ...
2. Chargeur de démarrage (GRUB / LILO)
3. Mode texte
4. Authentification par login + mot de passe
5. Shell (en mode console)

Changement de mode : **Ctrl** + **Alt** + **F1-F6/F7**

Ligne de commande vs interface graphique

- Inconvénients de la ligne de commande

- Avantages de la ligne de commande

Ligne de commande vs interface graphique

- Inconvénients de la ligne de commande
  - apprentissage plus long
  - efficacité moindre (utilisateur débutant)
  - mémorisation nécessaire (partiellement)
  - domaine d'application limité (mais pas tant que ça...)
- Avantages de la ligne de commande

Ligne de commande vs interface graphique

- Inconvénients de la ligne de commande
  - apprentissage plus long
  - efficacité moindre (utilisateur débutant)
  - mémorisation nécessaire (partiellement)
  - domaine d'application limité (mais pas tant que ça...)
- Avantages de la ligne de commande
  - automatisation aisée
  - efficacité (rapidité) supérieure (utilisateur aguerri)
  - ressources négligeables (CPU, réseau...)
  - expressivité plus forte (options)
  - modularité et extensibilité (une tâche, un outil)
  - compréhension et contrôle des actions

Session utilisateur

Comptes utilisateurs :

- session : login/mot de passe *(username/password)*
- homedir : répertoire personnel
- permissions d'accès aux ressources (fichiers, processus) :
  - utilisateur
  - groupe
  - autres

Un compte unique d'administrateur (super-utilisateur) : root

Des comptes "services"

- pour les tâches système : mail, impressions, ...
- des droits restreints (par rapport à root)
- sécurité accrue en cas de bug ou compromission

Découverte du shell - 1

103.1

Le prompt (invite de commandes)

- utilisateur courant
- nom de machine
- répertoire courant
- $ ou # : terminateur
- ... configurable à l'extrême
- un curseur !

Découverte du shell - quelques commandes

103.1

id	Qui suis-je ?
pwd	Où suis-je ?
uname -a	À qui ai-je l'honneur ?
lsb_release -a	Mais encore ?

ls	Liste les fichiers
cd	Changement du répertoire courant
man	Page de manuel d'une commande
cat	Affiche le contenu d'un fichier

Commandes : syntaxe générale *103.1*

Syntaxe :

```
commande [options] [- -] [paramètres]
```

Exemples :

- `ls --help`
- `ls -a`
- `ls --all`
- `ls -al`
- `ls -l .bashrc`
- `ls -w 60`
- `ls -w60`
- `ls --width=60`

Remarques : quelques exceptions

- `find . -name '*.tex' -print`
- `dd if=/dev/hda1 of=hda.img bs=512`

Commandes internes et externes *103.1*

- Commandes d'identification
  - `which` : commandes externes (fichiers)
  - `type (-a)` : commandes connues du shell
  - `whereis` : binaire et page de man d'une commande
- Les principaux types de commandes
  - **commande externe (fichier exécutable) - hashed**
  - **commande interne ou primitive shell (builtin)**
  - alias
  - fonction shell
  - mot-clé du shell, ex. `if`, `for`
- Exo : déterminer le type des commandes suivantes

  `cd, cp, ls, which, type, echo`

Documentation - formats et logiciels *103.1*

- aide en ligne de commande
  `ls --help`
- aide de bash : `help` (commandes internes)
- pages de manuel : `man`
  cf page suivante
- `info` : la documentation GNU (voir aussi `pinfo`, `tkinfo`)
- et encore : des pages .html, des fichiers `README`, `.chm`...
  voir `/usr/share/doc/`
- navigateurs d'aide (Gnome, KDE...) : interne, man, info...

Documentation - manpages *103.1*

- `man ls`, `man man`

- Neuf sections

1. commandes util.	4. périphériques	7. “conventions”
2. appels noyau (C)	**5. fichiers conf.**	**8. commandes admin.**
3. appels bibli. (C)	6. jeux	9. routines noyau

  `man (1) man`, `man 7 man`

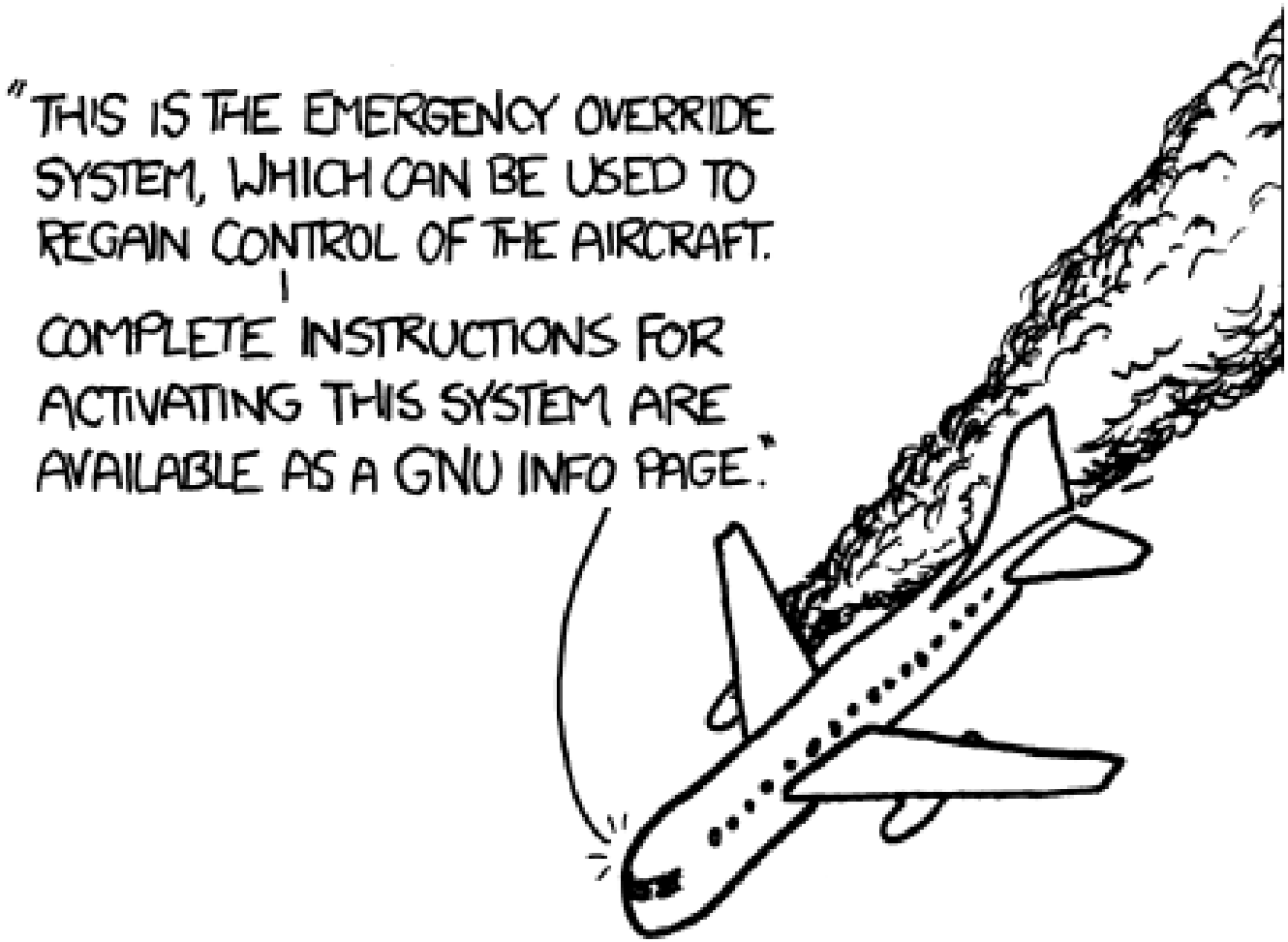
- Parties génériques : Nom, Synopsis, Description, Auteurs, Voir aussi...

- Pager `less` intégré : défilement
  - recherche : `/`motif, `n`, `N`, ...
  - marqueurs : `m`a ..., `'`a ...

- survivant du système roff/nroff/groff (formatage à balises)

XKCD 912 - Manual Override

103.1



http://xkcd.com/912/

Gestion des fichiers et répertoires *103.3*

- Commandes courantes
  - informatives : `ls, cat`
  - modificatrices : `touch, cp, mv, rm`
  - répertoires (informatives) : `pwd, cd, du, tree`
  - répertoires (modificatrices) : `mkdir, rmdir`
- Spécificités Unix
  - métadonnées Unix : `stat`
  - propriétaires : `chown`, `chgrp`
  - permissions : `chmod`
  - liens : `ln (-s)`, `readlink`

Récapitulatif : chemins relatifs et absolus *103.3*

- Chemins absolus : exemples
  - `ls /home/stg1/Linux`
  - `ls ~stg1/Linux`
  - `ls ~/Linux`
- Chemins relatifs : exemples
  - `ls Linux`
  - `ls ./Linux`
  - `ls ../AutreRepertoire`
- Ne pas confondre : fichiers et répertoires cachés
  ex. `ls -l ./.bashrc`

Globbing (expansion des noms de fichiers) *103.3*

But

Ne pas avoir à taper le nom de tous les fichiers en argument.

Exemple

```
ls *.rc
```

Caractères spéciaux

- `*` Tout
- `?` Un caractère quelconque
- `[a-z]` Un caractère parmi ceux listés

Protections contre l'interprétation par le shell

- `"..."` Protège partiellement ... de l'interprétation par le shell
- `'...'` Aucune interprétation de ...
- `\...` Aucune interprétation du caractère suivant

Pour aller plus loin : globbing personnalisé *103.3*

- Personnalisation du globbing
  - Commande shell `shopt (-s | -u) option`
  - Variable d'environnement : `$GLOBIGNORE`

- Options concernant le globbing

dotglob	inclut les fichiers "cachés"
failglob	erreur si rien ne correspond
globstar	récursif avec ** et **/
nocaseglob	insensible à la casse
nullglob	chaîne vide si rien ne correspond
extglob	motifs étendus

Pour aller plus loin : globbing personnalisé *103.3*

- Personnalisation du globbing
  - Commande shell `shopt (-s | -u) option`
  - Variable d'environnement : `$GLOBIGNORE`

- Options concernant le globbing

dotglob	inclut les fichiers "cachés"
failglob	erreur si rien ne correspond
globstar	récursif avec `**` et `**/`
nocaseglob	insensible à la casse
nullglob	chaîne vide si rien ne correspond
extglob	motifs étendus

Gestion des répertoires *103.3, 104.2*

- Rappels
  - `cd <cible>` changer de répertoire courant
  - `pwd` afficher le répertoire courant
- Création
  - `mkdir Toto`
  - `mkdir [-v] -p Toto/Titi/Tata/Tutu`
- Suppression
  - `rmdir [-p] <cible>`
  - `rm -rf <cible>` si les répertoires sont peuplés !
- Place disque occupée (104.2)
  - `df [-k|-m|-h] [-t ext3] ...` (*disk free*) tailles partitions
  - `du [-k|-m|-h] [-s] <cible>` (*disk usage*) tailles répertoires

Gestion courante des fichiers *103.3*

- Suppression : `rm [-r] [-f|-i|-I] [-v] <cibles>`
- Copie : `cp [-r] [-f|-i|-n] [-v] <source> <destination>`
  ou `cp [...] -t <rép-destination> <sources>`
- Déplacement : `mv [-f|-i|-n] [-v] <source> <destination>`
  ou `mv [...] -t <rép-destination> <sources>`
- Renommage : `mv [-f|-i|-n] [-v] <source> <destination>`
  - renommage simple (emplacement fixe)
  - renommage avec déplacement
- Pour aller plus loin
  - `mmv` (et dérivées) renomme fichiers multiples d'après un motif
  - `rename` expressions régulières et réécriture (perl)

Propriétaires d'un fichier *103.3*

Commandes principales

- `chown user:group file`
- `chgrp group file`
- user : `uid` / username
- group : `gid` / groupname

Commandes auxiliaires

- `groups` : à quels groupes appartient l'utilisateur
- `su` : changer d'utilisateur
- `adduser user group`

Permissions sur les entrées de répertoires *103.3*

Trois cibles de permissions

- u=user : utilisateur propriétaire
- g=group : groupe propriétaire
- o=other : tous les autres
- (a=all : tout le monde)

Trois types de droits

	sur fichier	sur répertoire
r=read	lecture	listage
w=write	écriture	ajout/suppression fichier
x=exec	exécution	traversée
X=exec	*conditionnelle*	*traversée*

Modifier les permissions *103.3*

Commande `chmod`

1. `chmod u=rwx,g=rx,o= <fichier>`
2. `chmod u+w,a+x <fichier>`
3. `chmod -R g=u`

Notation octale

- r=4, w=2, x=1 ex. `rwx r-x r--` = 754
- `chmod 750 <fichier>`

Permissions : Travaux pratiques *103.3*

Exercice : Remise des devoirs

Un enseignant cherche à récolter les programmes rédigés par ses étudiants dans un répertoire commun. Tous doivent pouvoir déposer un fichier, mais aucun ne doit pouvoir lister ni lire les autres fichiers déposés.

- Mettre en place la configuration nécessaire, ouverte à tous les utilisateurs.
- Comment restreindre le dépôt à un groupe de TP, nommé `tp01` ?
- Comment éviter les conflits de nommage entre plusieurs étudiants ?

Permissions Unix - Compléments *103.3*

SUID et SGID

- suid : changement d'UID à l'exécution `chmod u+s fichier`
- sgid : changement de GID `chmod g+s fichier`

Sticky bit

- fichier : obsolète
- répertoire : restriction à l'ajout/suppression d'entrées
  `chmod +t rép.`

- Extension ACL : Access Control List `man 5 acl`
- `man chmod`

Métadonnées Unix *103.3*

- Commande `stat` : sur fichier ou système de fichiers
- Permissions
  - utilisateur propriétaire : `uid` numérique
  - groupe propriétaire : `gid` numérique
  - `mode` r,w,x... (champ de bits) ex. `0644/-rw-r-r-`
- Horodatage
  - `atime` (access) : dernier accès (lecture) `ls -lu`
  - `ctime` (change) : modification des métadonnées (inode) `ls -lc`
  - `mtime` (modification) : modification du contenu `ls -l`
  - `touch` : mise à jour ⇒ atime, mtime falsifiables, ctime sûr !

  Exo : que devient l'horodatage en cas de : `cat`, `vim` (avec et sans modif), `mv` (renommage), `chmod` ?
- Auxiliaires
  - type de fichier (régulier, répertoire...)
  - taille en octets
  - compteur de liens

Liens physiques et liens symboliques - en pratique *103.3*

```
$ touch fichier
$ cp fichier fichier-cp
$ ln fichier fichier-ln      #lien physique
$ ln -s fichier fichier-lns  #lien symbolique

$ ls --inode --long

2080774 -rw-r--r-- 2  [...] fichier
2080775 -rw-r--r-- 1  [...] fichier-cp
2080774 -rw-r--r-- 2  [...] fichier-ln
2080776 lrwxrwxrwx 1  [...] fichier-lns -> fichier

$ ln -s fichier-lns fichier-lns2
$ readlink fichier-lns2
$ readlink -f fichier-lns2
```

Liens physiques et liens symboliques - inodes *103.3*

Usages des liens symboliques

- Alternatives ex. `vim -> /usr/bin/vim.basic`
- Rétro-compatibilité ex. `/tmp -> /var/tmp`
- “Raccourcis”
  ex. `./doc -> /usr/share/doc/debian-reference-fr`

Usage des liens physiques

- relativement obsolète
- "instantané" (snapshot), cf `rsnapshot`

Liens physiques et liens symboliques - inodes *103.3*

Structure du système de fichiers - inodes

- répertoires
- inodes (métadonnées)
- contenus

Liens physiques et liens symboliques - comparaison *103.3*

	lien symbolique	**lien dur**
pointe sur	entrée de répertoire	inode
rôle	asymétrique	symétrique
cible	tout type	fichier régulier
chemin cible	absolu ou relatif	N.A. (inode)
système de fichiers	interne ou externe	interne
cohérence	peut être cassé	jamais cassé
stockage	fichier (spécial)	entrée de répertoire

Archives et compression *103.3*

- L'archivage : rassembler plusieurs fichiers en un seul.
  `tar -c`, `tar -x`, `tar -t`
  `-f archive.tar` : spécifier le fichier archive (sinon flux)

- La compression
  - `gzip` + `gunzip` (ou `tar -z ...`)
  - ...

- Exercice
  1. Prendre connaissance du contenu de `tp-access.tgz`
  2. Décompresser l'archive
  3. Créer une archive compressée avec les 20 premiers fichiers
  4. Compresser individuellement les 20 derniers

- Autres implémentations : S-tar (`star`)...
- Unix historique : `cpio` + `compress` (.Z)

Compression *103.3*

- Utilitaires et algorithmes de compression
  - `gzip` + `gunzip` (ou `tar -z`), `zcat`, `zless` ... algo LZ77
  - `bzip2` + `bunzip2` (ou `tar -j`), `bzcat` ... algo Bzip2
  - `xz` + `unxz` (ou `tar -J`), `xzcat` ... algo LZMA
  - `compress` + `uncompress` (ou `tar -Z`) obsolète, algo LZW

- Travaux pratiques
  1. *benchmarking* des temps et tailles entre les compressions `gzip`, `bzip2` et `xz`.
  2. autres variables : niveaux de compression des algorithmes, taille mémoire...

- Compatibilité Windows : `zip`, `7z` (paquets `zip` et `p7zip-full`)

Rechercher un fichier... 1/2 Indexation *104.7*

- `locate` : recherche rapide dans une base de données
  - `locate` (GNU) : source `findutils`
    - `-r` expression régulière, ex. `-r fst.b`
    - `-S` statistiques ...
  - `slocate` (obsolète) : + permissions
  - `mlocate` : + optimisation base

  - TP : Avec `updatedb` : lancer une indexation personnelle de son répertoire

- Fichiers et paquets (distribution)
  - (Debian) `dlocate` : recherche parmi les paquets installés alternative rapide à `dpkg -S`
  - (RH) `rpm -qf`

Rechercher un fichier - 2/2 Find

104.7

- `find` : recherche multicritères
  `find /etc/ -size +10k -ctime -10 -printf '%s %p'`
  - répertoire de départ (/etc)
  - options de sélection (size, ctime)
  - options d'action (printf)

  - Toujours à jour
  - Potentiellement plus long que `locate`

- Exercices
  - Pour aller plus loin : options `find -H -L -P`
  - Trouver le nombre d'entrées de répertoire de chaque type sous `/`, sans changer de système de fichiers (`-xdev`).

  - Pour les quatre types minoritaires, afficher les entrées.

Récapitulatif : différences avec le système de fichiers de Windows

- Pas de notion de lecteur C: D: etc.
- Tout est dans une même arborescence, de racine /
- Les répertoires sont séparés par des / et non des \
- Existence de liens symboliques
  `ln -s fichier lien`
  Sous windows, les liens sont de simples fichiers *.link*
- Des permissions explicites

Principaux types de fichiers

Principaux types de fichiers

- Trois principales distinctions :
  - texte ou binaire
  - exécutable ou pas
  - installé par la distribution ou pas

Principaux types de fichiers

- Trois principales distinctions :
  - texte ou binaire
  - exécutable ou pas
  - installé par la distribution ou pas
- Quelques exemples :
  - programmes binaires, ex. `/bin/cp`
  - scripts shell, ex. `/etc/init.d/rc.local`
  - fichiers de configuration, ex. `/etc/fstab`
  - fichiers de log, ex. `/var/log/messages`
  - bibliothèques dynamiques **.so**

Principaux types de fichiers

- Trois principales distinctions :
  - texte ou binaire
  - exécutable ou pas
  - installé par la distribution ou pas
- Quelques exemples :
  - programmes binaires, ex. `/bin/cp`
  - scripts shell, ex. `/etc/init.d/rc.local`
  - fichiers de configuration, ex. `/etc/fstab`
  - fichiers de log, ex. `/var/log/messages`
  - bibliothèques dynamiques **.so**
- Commandes utiles
  - `file` : le type du fichier
  - `which` ou `type` : pour une commande
  - `cat, head, tail` : le contenu du fichier (texte)
  - `hd, ldd, strings...` : le contenu du fichier (binaire)

Exécutables interprétés et compilés *104.7*

- Langages interprétés
  - interpréteur standard : shell (`bash` ou ...)
  - autres : perl, python, ruby, php
  - *shebang* (ou *hashbang*) : `#! /usr/bin/perl -w`
  - interpréteur nécessaire pour l'exécution
  - code source = exécutable

- Langages compilés
  - entrée : code source texte ex. C, C++, Fortran...
  - chaîne de compilation : `gcc`, `as`, `ld`
  - sortie : binaire exécutable ELF (...)
  - source (C...) $\longrightarrow$ compilateur $\longrightarrow$ exécutable ELF
  - code source $\neq$ exécutable

ELF : Executable and Linkable Format *104.7*

Le format standard des exécutables Linux

- Buts
  - Assembler les unités de compilation (*.o)
  - Créer une image mémoire d'un programme

- Trois sous-types de fichiers ELF

  EXEC binaire exécutable
  DYN fichier objet partagé `*.so`
  REL fichier relocalisable `*.o, *.a`

- Commandes disponibles
  - `file /bin/ls` ⟶ ELF 32-bit LSB executable [...]
  - Pour aller plus loin : `readelf -h`, `nm`, `objdump`

Pour aller plus loin : file et MIME

Comment déterminer un type de fichiers ?

- Plusieurs concepts à distinguer
  - l'extension du fichier (si elle existe) : métadonnée
  - sa signature (si elle existe)
  - son type MIME (Multipurpose Internet Mail Extensions)
  - les applications le prenant en charge
- Techniquement
  - `libmagic` à la base de `file` : `man magic`
  - `file -i` renvoie le type MIME
  - `/etc/mime.types`
  - `/etc/mime-magic` et `/etc/magic`

Filesystem Hierarchy Standard 1/2

104.7

Norme FHS maintenue par la Linux Foundation

/	racine
/home/	répertoires utilisateurs
/root/	homedir de root

Filesystem Hierarchy Standard 1/2

104.7

Norme FHS maintenue par la Linux Foundation

/	racine
/home/	répertoires utilisateurs
/root/	homedir de root
/bin/	exécutables principaux (système)
/sbin/	exécutables d'administration (superuser)
/etc/	configuration du système

Filesystem Hierarchy Standard 1/2

104.7

Norme FHS maintenue par la Linux Foundation

/	racine
/home/	répertoires utilisateurs
/root/	homedir de root
/bin/	exécutables principaux (système)
/sbin/	exécutables d'administration (superuser)
/etc/	configuration du système
/usr/	programmes (gérés par la distribution)
/usr/bin/	exécutables des programmes
...	
/usr/local/	programmes (hors distribution)

Filesystem Hierarchy Standard 1/2 *104.7*

Norme FHS maintenue par la Linux Foundation

/	racine
/home/	répertoires utilisateurs
/root/	homedir de root
/bin/	exécutables principaux (système)
/sbin/	exécutables d'administration (superuser)
/etc/	configuration du système
/usr/	programmes (gérés par la distribution)
/usr/bin/	exécutables des programmes
...	
/usr/local/	programmes (hors distribution)
/var/	données variables
/var/log	fichiers de log
/var/spool	fichiers tampons (mail, impressions...)

Filesystem Hierarchy Standard 2/2 *104.7*

Extensions...

/opt	applications installées hors conventions Unix
/mnt	montages externes (réseau...)
/media	montages amovibles (CD, clé USB...)
/srv	données utilisées par les services (FTP, WWW...)
/selinux	réservé pour Security Enhanced Linux
/run	données runtime (remplace /var/lock et /var/run) (non-LSB)

Systèmes “virtuels” (tout est fichier...)

/dev	fichiers-périphériques
/proc	informations sur les processus : `man 5 proc`
/sys	informations système

Points de montage (introduction) *104.7*

Comment accéder à un CD-ROM sans D: ?

Points de montage (introduction) *104.7*

Comment accéder à un CD-ROM sans D: ?

`mount /media/cdrom`

Les points de montage

Initialement, seule existe la racine /.

Puis `mount` sert à associer

- un périphérique physique (disque, partition) ex. `/dev/sda2`
- un répertoire ex. `/mnt/windowsC`

Exemple : `mount -t vfat /dev/sda2 /mnt/windowsC`

Les montages par défaut sont décrits dans `/etc/fstab`.

`mount` (sans argument) liste les montages en cours.

Pour aller plus loin

- automontage : clés USB, périphériques *hotplug*
- montage par l'interface graphique

Redirections - canaux *103.4*

Le shell définit 3 canaux

STDIN (0) entrée standard - clavier par défaut

STDOUT (1) sortie standard - écran (terminal) par défaut

STDERR (2) sortie d'erreur - écran (terminal) par défaut

Redirection

```
ls -1 > liste.txt
```

La sortie du programme `ls` est **redirigée** vers un fichier.

Pour **ajouter** au fichier (sans écraser l'ancien contenu) :

```
ls -1 >> liste.txt
```

`2>` redirection de la sortie d'erreur

`&>` redirection des deux sorties

`<` redirection d'entrée, ex. `cat < liste.txt`

Pipes et filtres

103.4

`ls -1 | wc` sortie de `ls` canalisée vers l'entrée du filtre `wc`.

`find /etc |& wc` StdOut et StdErr fusionnées puis canalisées

Exemples

1. `cat` taper `Ctrl+D` = fin de flux
2. `cat liste.txt | wc -l`
3. `wc -l liste.txt`
4. `wc -l < liste.txt`
5. `cat < liste.txt | wc -l`
6. `wc -l liste.txt l2.txt l3.txt`
7. `cat liste.txt l2.txt l3.txt | wc -l`

Exo. Dessiner le schéma correspondant à chacune des commandes. Identifier filtres et semi-filtres.

Filtres textes courants *103.2*

Principe Unix : une tâche, un outil.

Beaucoup de filtres fonctionnent ligne par ligne :

- `head` Premières lignes
- `tail` Dernières lignes
- `sort` Trie les lignes
- `uniq` Enlève les doublons
- `grep` Garde les lignes correspondant à une expression donnée.
  Ex. `ls / | grep v`
- `cut` Conserve les colonnes (resp. champs) donnés

- moins courants : `tr`, `tac`, `paste`, `fmt`...
- paquet `coreutils`

TP : synthèse de logs

103.2

Le fichier `access.log` contient un extrait de logs du serveur Apache, duquel on va essayer de tirer des statistiques.

1. Combien de requêtes sont enregistrées dans le fichier access.log ?
2. Extraire du fichier access.log la liste des adresses IP clientes.
3. Compter le nombre d'adresses IP différentes.
4. Afficher le nombre d'occurrences de chaque IP, puis présenter la liste par nombre décroissant d'occurrences.
5. Afficher uniquement les IP ayant effectué au moins 10 accès.
6. Question subsidiaire : pour chacune des IP de la liste précédente, effectuer une résolution de nom (commande host).
   a. en passant par un fichier temporaire
   b. sans intermédiaire, en une seule ligne de commande

TP : manipulation de texte *103.2*

Le fichier `auteurs.txt` contient une liste d'auteurs avec leur fréquence d'apparition. Ceux qui sont placés entre « ... » sont identifés clairement, à la différence des autres.

1. Séparer énoncé et données dans deux fichiers différents.
2. Combien y a-t-il d'auteurs au total ? Combien de bien identifiés ? De mal identifiés ?
3. Classer les auteurs selon leur fréquence.
4. Lister les 20 auteurs les plus courants, le plus fréquent en premier.
5. Créer un fichier `auteurs2.txt` dans lequels ne figurent pas les auteurs n'ayant qu'une occurrence. Combien sont-ils ?
6. Quels sont les 10 auteurs mal identifiés qui apparaissent le plus souvent ?

Pour aller plus loin : Sed et Awk

103.2

- Sed : Stream Editor
  - adapté aux opérations sur les chaînes et les regexp
  - `sed -e "s/Old/New/g" f-in > f-out`
- AWK : un langage-filtre
  - pour les fichiers structurés en colonnes ou en champs
  - `gawk -F: '$3 > 999' /etc/passwd`
- Encore plus loin : Perl

Filtres - pour aller plus loin *103.2*

- La commande `tee` : brancher une dérivation
  ```
  egrep ":[0-9]:" /etc/passwd | tee listing | wc -l
  ```
- La commande `xargs` : transformer STDIN en arguments
  ```
  find /etc/ -size +100k | xargs wc -l
  ```
- La commande `mkfifo` : créer un pipe nommé
  ```
  mkfifo listing
  cut -d: -f1-3 listing
  egrep ":[0-9]:" /etc/passwd | tee listing | wc -l
  ```
  $\longrightarrow$ synchronisation forcée des processus

Pour aller plus loin : fonctions avancées du shell

- Mode interactif
  - autocompletion
  - raccourcis clavier
  - historique
  - alias
- Scripts shell
  - gestion des arguments
  - boucles
  - tests et structures de contrôle (if ...)
  - fonctions
- Configuration du shell
- Choix du shell : `bash`, `tcsh`, `zsh`...

Processus et tâches *103.5*

Gestion des tâches (jobs)

- `commande &` : lancer en arrière-plan
- `jobs (-l)`
- **Ctrl-Z** : met en pause
- **Ctrl-C** : arrête
- `bg` : redémarre en arrière-plan le processus en pause
- `fg` : remet en avant-plan

Affichage des processus

- `top` : affiche les ressources consommées par les processus
- `ps` : Process Show
- `pstree` : **arbre** des processus $\longrightarrow$ init
- `prtstat` : (paquet `psmisc`) tous les détails d'un processus
- `qps` : interface graphique conviviale

Commande ps - les options *103.5*

1. syntaxe BSD : `ps U root`, `ps aux`
2. syntaxe SysV : `ps -U root`, `ps -ef`
3. syntaxe longue GNU : `ps --user root`

Principales options

1. Options de sélection
   - -e, -A : tous les processus
   - -C <liste commandes>
   - -G, -U ... <liste utilisateurs, groupes>
   - -t, --tty <liste de terminaux>
2. Options de niveau de détail
   - -f, -F : full, extra-full
   - -o, -O, --format : personnalisé Ex. `ps -O ppid,pgrp,sess`
3. Options d'affichage
   - --sort : tri Ex. `--sort tt,-pid`
   - -H, --forest : hiérarchie
   - --headers --lines=20 : répéter l'en-tête toutes les 20 lignes

Regroupement de processus 1/2 - sessions *103.5*

- **session (SESS)** : processus d'une session
- *session leader* (bash...) : fournit son PID à la session
- **TPGID** : groupe au premier plan du terminal (TTY) du processus
- `ps -t pts/1 -O ppid,sess`

Terminaux et pseudo-terminaux

- Consoles virtuelles (TTY)
  - consoles texte standard (Alt + F1-F6...)
  - `/dev/tty1-63`
- Pseudo-terminaux (PTYs)
  - terminaux X, session shell...
  - `/dev/pts/0...` + `/dev/ptmx` (System V)

Regroupement de processus 2/2 - groupes *103.5*

Regroupements de processus

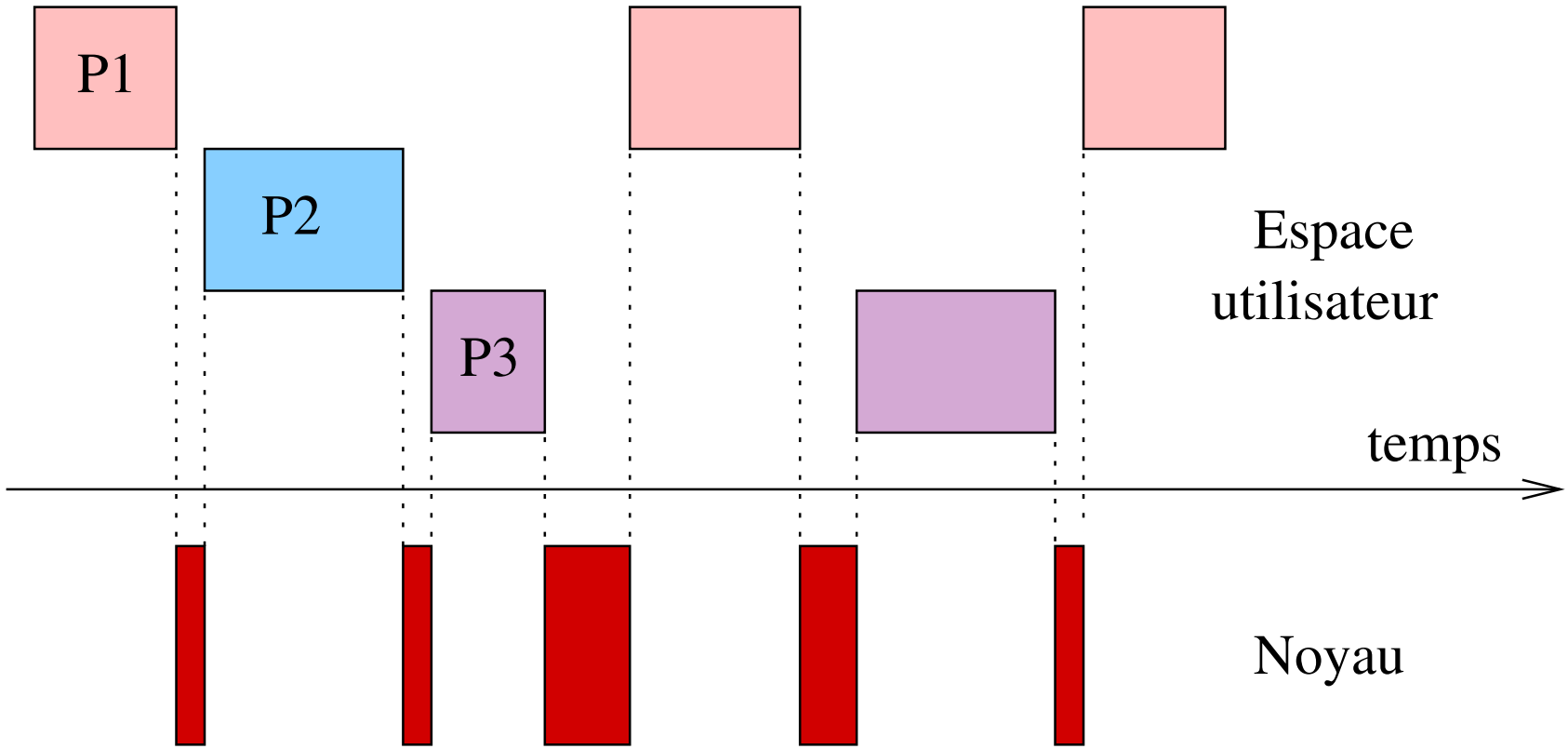
- **groupe (PGRP)** : processus formant une même commande (=job)
  ex. `find / | grep pass | less`
  group leader = `find`
- une *session* regroupe plusieurs *groupes*

Exercice

- Combien de sessions différentes et de groupes différents tournent sur la machine ?

Notions d'ordonnancement

103.6



Paramètres

- fréquence : réactivité du système
- proportion : priorités des processus (cf *niceness*)

Processus - états et ordonnancement *103.6*

Etats des processus

R	demande d'exécution (Running)
S	attente interruptible (Sleep)
D	attente non interruptible (I/O)
T	stoppé (par SIGSTOP)
Z	"zombie" (ou defunct)

<	priorité haute
N	priorité basse (Nice)
s	session leader
l	multi-thread
+	groupe d'avant-plan

Trois classes d'ordonnanceur (CLS)

- TS : Time Shared (standard)
- FF : Real Time Fifo
- RR : Real Time Round Robin

Charge machine

103.6

Commandes

- `w`, `uptime` (statique)
- `top` (dynamique)
- `xload` (graphique basique)

Définition de la charge

- Nombre moyen de processus dans l'état exécutable (R) ou en attente non interruptible (D).
- Moyenne temporelle sur 1, 5 et 15 minutes.

Priorité et "courtoisie" *103.6*

Courtoisie (*niceness* NI)

- un nombre entier, entre -20 et 19
- -20 à -1 : réservé à root, priorités hautes
- 0 : valeur par défaut
- 1 à 19 : accessibles à tous, priorités basses

Priorité (PRI) : calculée à partir de la courtoisie

- $PRI = 19 - NI$ en temps partagé
- $PRI = 19 - NI + 100$ en temps réel (FF, RR)

Commandes

- nice commande *Ex : nice -n10 md5sum cd.iso*
- renice courtoisie PID *Ex : renice +20 5124*

Contrôle des processus et signaux *103.5*

Rechercher un processus

`pgrep` : recherche multicritères

Arrêter un processus

- `kill` [options] PID
- `killall` commande
- `pkill` [-signal]

```
kill -TERM 1955
killall gimp
```

Les principaux signaux

- SIGTERM (15) : terminer normalement (“proprement”)
- SIGKILL (9) : terminaison forcée (non ignoré)
- SIGSTOP (19) : arrêt temporaire (pause) (non ignoré)
- SIGCONT (18) : reprise d'un processus arrêté
- SIGINT (2) : terminaison interactive (Ctrl-C)
- SIGTSTP (20) : arrêt temporaire interactif (Ctrl-Z)

Pour aller plus loin : threads noyau *103.5+*

Les threads noyau

- le démon `kthreadd (PID=2)`
- et tous ses fils : `ps -f --ppid=2`
- parfois liés à un processeur : `[ksoftirqd/0]`

En pratique

- Combien de threads noyau sont en cours d'exécution ?
- Quel est le premier "vrai" processus utilisateur ? (hors init)

Processus légers (threads) *103.5+*

Les threads : des “sous-processus”

Partage de : **code**, données, E/S fichiers, signaux, *pile*

Les threads utilisateurs : affichage avec `ps`

- `ps -L -f` : LWP (pid du thread), NLWP (nombre de threads)
- `ps -Lf -m` : sous-processus affichés après les processus “principaux”

En pratique

- Combien de processus multi-threadés tournent ?
- Combien de threads au maximum ? Pour quel processus ?

TP – Processus

103.5

1. Combien, approximativement, de processus ont été créés depuis le dernier démarrage du système ?
2. Lister les processus **bash** en cours.
3. Utiliser `top` pour trouver le processus utilisant le plus de mémoire. Tenter de l'arrêter.
4. Faire le lien entre `/proc/` et les processus. Cf `man 5 proc`

5. Trouver le processus de PID maximal, puis le dernier processus lancé
6. Chercher le taux de création des processus (en p/s).
7. Créer une fonction pour rechercher le père d'un processus donné, puis une autre pour déterminer la profondeur d'un processus donné (en argument)

Pour aller plus loin...

- Surveiller un processus avec `watch`
  ```
  watch ls -l /var/log/messages
  watch -d ps -F
  ```
- Utiliser `wait` (interne) pour synchroniser des tâches (script)
- Utiliser `procinfo`
- Utiliser `unhide` pour chercher les processus dissimulés (rootkits...)
- Utiliser `pidstat` pour obtenir les ressources utilisées (paquet `sysstat`)

Éditeurs de texte

103.8 ?

Éditeurs sans interface graphique

- parfois nécessaire (connexion réseau, problème graphique)
- plus rapide

Éditeurs de texte

103.8 ?

Éditeurs sans interface graphique

- parfois nécessaire (connexion réseau, problème graphique)
- plus rapide

- nano
  - simple d'utilisation
  - installé par défaut
- emacs -nw
  - puissant et configurable
  - généralement utilisé en mode graphique
- vi / vim
  - éditeur modal : déroutant au premier abord
  - puissant et efficace pour l'administration système

vi / vim *103.8*

- Historique Vi
  - `qed` → `ed` (K. Thomson) → `ex` → `vi`
  - 1976 par Bill Joy, étudiant à Berkeley (puis `csh`, `NFS`, Sun)
  - mode *visuel* de `ex` : premier éditeur "pleine page"
  - POSIX (IEEE 1003.2, Part 2 : Shell and utilities)
  - Développement stoppé en 1985 (licence Sun)

- Nombreuses variantes
  - `elvis`, Steve Kirkendall (Minix, Slackware), 1990-2003 ?
  - `nvi`, Keith Bostic (4.4BSD et dérivés libres), 1992-1996 ?
  - `vile` : VI Like Emacs

- VIM (Vi IMproved)
  - auteur Bram Moolenar (NE)
  - 1991 (1.0) - 2008 (7.2)...
  - toutes plateformes : Unix, Linux, Windows...
  - interfaces graphiques : gtk et gnome

vim - en pratique *103.8*

Fonctionne par modes : commande, édition, visualisation.

Raccourcis principaux

Esc	sortir du mode courant
i	insérer (insert)
yy	copier une ligne (yank)
dd	coupe une ligne (delete)
p	coller (put)
:w	écrire dans le fichier (write)
:q	quitter vim (quit)

Pour aller plus loin

- 5dw $\longrightarrow$ efface 5 mots
- yf, $\longrightarrow$ copie le texte jusqu'à la prochaine virgule
- vimtutor pour s'entraîner aux manipulations

vim - Fichiers de configuration *103.8*

Fichiers de configuration

- `/etc/vim/vimrc` : global système
- `~/.vimrc` : personnel, ex. :

```
syntax on
set nu
```

Fichiers auxiliaires

- `~/.viminfo` : historique commandes, tampons . . .

Emacs -

- Historique
  - 1976 : TecoEmacs, Steele et Stallman (MIT) sur PDP/ITS
  - 1978 : MulticsEmacs (B. Greensberg), Lisp langage d'extension
  - 1981 : GoslingEmacs (J. Gosling), 1ère version Unix
  - 1984 : intégré au projet GNU, réécrit (R. Stallman)
  - 1985 : **GnuEmacs 15.34**, 1ère version largement diffusée

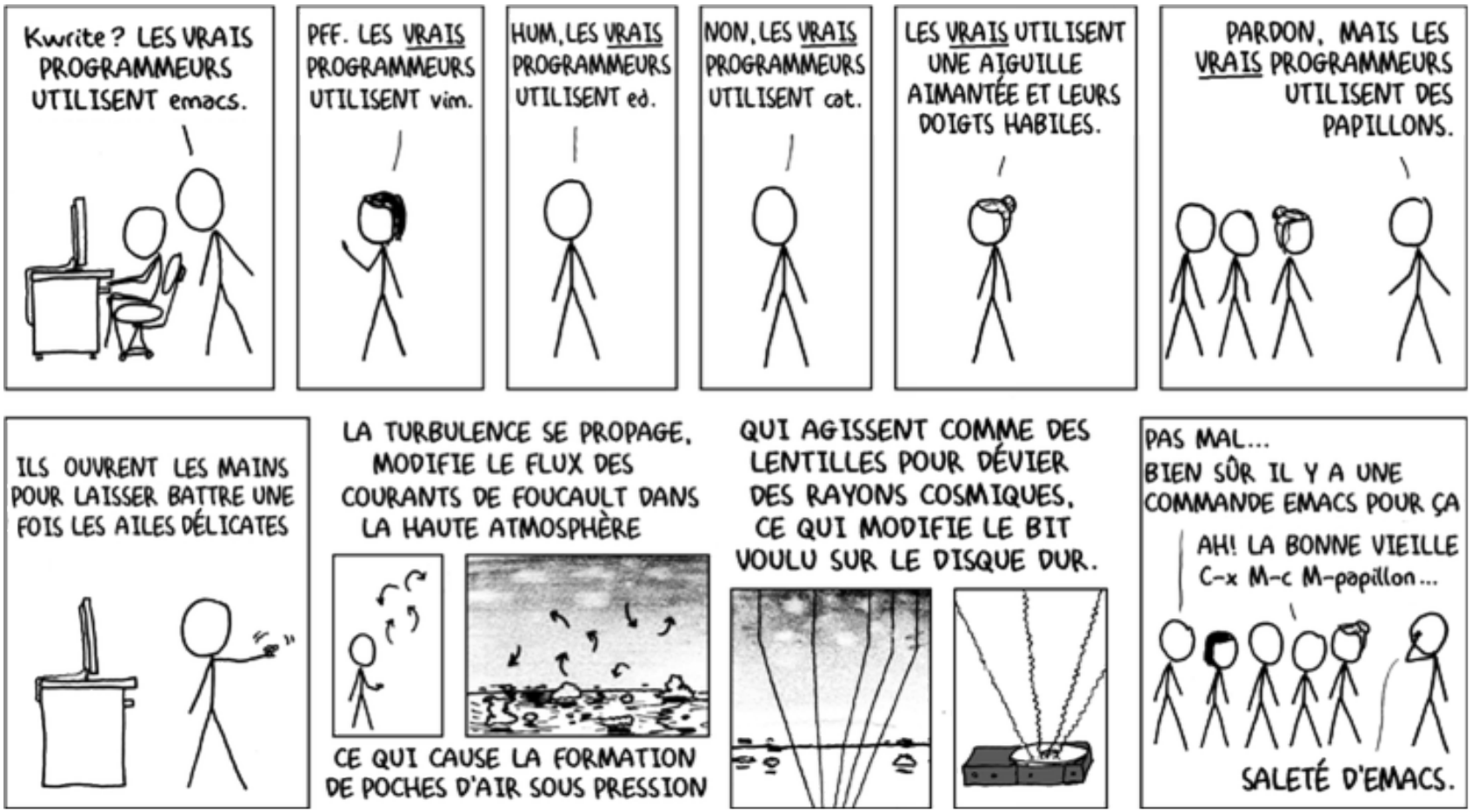
  ...
  - juin 2007 : GNU Emacs 22.1
  - sep. 2008 : GNU Emacs 22.3
  - Emacs 23 en préparation (Unicode natif)
  - voir `http://www.jwz.org/doc/emacs-timeline.html`

- Variantes
  - XEmacs (1991-) Lucid Inc.
  - MicroEmacs, plus compact
  - ...

Emacs - en pratique -

- Fonctionnalités
  - fonctionnement “moderne” (monomode)
  - implémenté en langage C
  - extensions en Emacs Lisp (eLisp)
- Trois modes de configuration
  - extension *Customize* (menus, GUI)
  - enregistrement de macros
  - utilisation d'eLisp et fichier `.emacs` ou `.emacs.d/*`

XKCD 378



(C) Randall Munroe, CC-BY-NC, trad. P. Gambette

http://www.lirmm.fr/~gambette/xkcd/index.php?id=378

Compilation d'un exécutable

- Exemple : compilation de `ncdu`
- Procédure standardisée : utilisation d'autoconf/automake
  - `./configure (--help)`
  - `make`
  - `make install`
- Dépannage : recherche de dépendances (bibliothèques dynamiques)

Diff et Patch

- Commande `diff`
  - direct : entre deux fichiers
  - -c, -u : contexte, unifié
  - -r : récursif (entre répertoires)
- Commande `patch`
  - syntaxe `patch -p0 <patchfile`

Architecture TCP/IP

109.2

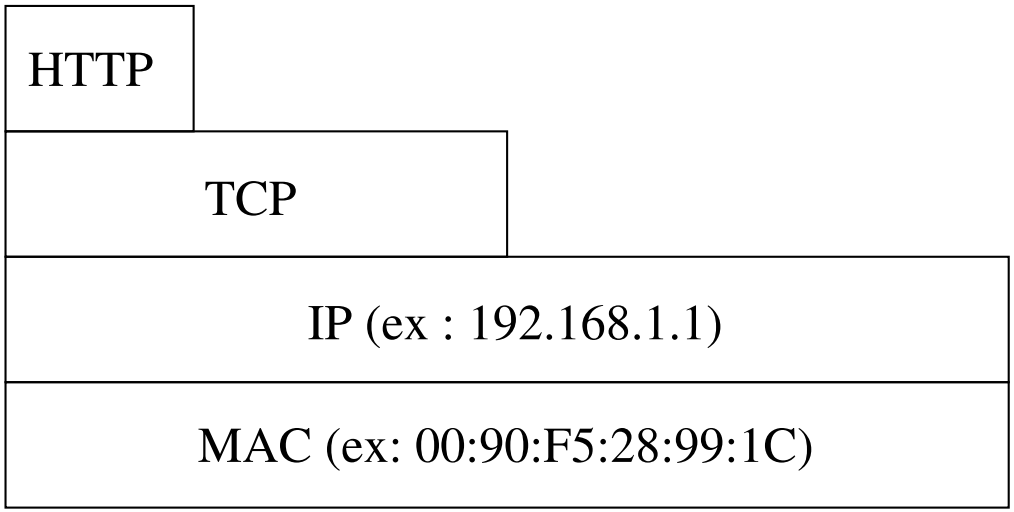
Un modèle par couches

ernet réseau local Ethernet-MAC

IP l'adressage Internet, avec une double fonction

- identifiant unique de l'hôte sur le réseau (*identifier*)
- emplacement sur le réseau (topologie) (*locator*)

TCP le transport



Commandes de diagnostic *109.2*

ifconfig

- lo *(interface virtuelle boucle locale)*
- eth0 *(première interface ethernet)*

- l'adresse MAC : 6 octets
  ex. *HWaddr : 00 :90 :F5 :28 :99 :1C*
  Propre à la carte réseau
- l'adresse IPv4 : 4 octets, 32 bits
  ex. *inet addr : 192.168.1.1*
- l'adresse IPv6 : 16 octets, 128 bits
  ex. *inet6 : fe80 : :219 :66ff :fee9 :381/64*

Commandes de diagnostic - 2 *109.2*

- `ping` Tester soi-même, un voisin, un absent, le réseau...
  - `ping -a -c5 192.168.1.1`
  - `ping -b 192.168.1.0`

- `traceroute` (champ TTL)
  affiche le chemin suivi par un paquet (tous les routeurs)

- `mtr` (my traceroute)
  combinaison des deux précédentes commandes

Résolution de noms (DNS) *109.4*

En local : /etc/hosts

Établit des correspondances *nom d'hôte* $\leftrightarrows$ *adresse IP*

Domaine Name Server (DNS)

- Permet une équivalence entre nom et adresse IP.
  Ex : coriolan.silecs.info $\leftrightarrows$ 82.233.121.16
- Fonctionnement par hiérarchie de serveurs

Clients DNS

- Client léger : `nslookup`
- Clients complets :
  - `dig` (dnsutils)
  - `host` (host)
- Sans oublier `ping` (/etc/hosts puis DNS)

Exemple de service : SSH

SSH : connexions sécurisées *110.3*

La famille SSH

- `sshd` : le serveur
- Les clients essentiels
  - `ssh`, `slogin` : connexion interactive ou batch
  - `scp` : copie de fichiers via ssh
  - `sftp` : émulation ftp via ssh
- Les utilitaires
  - gérer les clés utilisateurs : `ssh-keygen`, `ssh-copy-id`
  - mémorisation des clés : `ssh-agent`, `ssh-add`

Remarques

- conçu pour remplacer `rlogin`, `rcp`...
- X11 forwarding : ouverture à distance d'applis graphiques

Clients SSH - 1 - shell distant *110.3*

- Shell interactif `slogin`
  - `slogin user@distant`
  - Variables d'environnement : `env | grep SSH` : `SSH_CLIENT`, `SSH_TTY`, `SSH_CONNECTIONS`
  - Qui est là ? commandes `who` et `w -l`
- X11 Forwarding
  - `slogin -X | -Y user@distant`
  - Variable d'environnement `DISPLAY=localhost:10.0`
- Shell non-interactif (commande à distance) `ssh`
  - `ssh user@distant /bin/ls`
  - `ssh user@distant "cat /etc/passwd | grep /home"`
  - `ssh user@distant "cat /etc/passwd" | grep /home`

Clients SSH - 2 - transferts de fichiers *110.3*

- Copie distante `scp`
  - `scp user@distant:/home/user/.bashrc ./bashrc` *pull*
  - `scp ./fichier.txt user@distant: /Linux/` *push*
- Protocole SFTP (SSH File Transfer Protocol)
  - `sftp user@host:/path/to/dir` puis session interactive
  - `lftp` ou autres commandes multi-protocoles
  - graphique : gftp, filezilla, ou autres interfaces multi-protocoles
  - Note : SFTP $\neq$ FTPS (FTP over SSL) !
- TP pour aller plus loin : copie réseau en flux avec `tar` et `ssh`.

Cryptographie symétrique et asymétrique *110.3*

Chiffrement symétrique

Une seule clé pour le chiffrage et le déchiffrage

Chiffrement asymétrique

- Principe
  - une clé privée + une clé publique
  - une clé chiffre, l'autre déchiffre
  - secret : chiffrement avec la clé publique du destinataire
  - authentification : chiffrement avec la clé privée de l'expéditeur
  - une infrastructure de distribution des clés publiques (PKI)
- Diversité des clés SSH
  - clés d'hôtes (systématiques) vs clés d'utilisateur (optionnelles)
  - clés RSA, DSA, ECDSA : trois algorithmes différents
  - clé publique vs privée

Authentification utilisateur SSH par biclé *110.3*

1. Création de la clé
   `ssh-keygen -t rsa -C "commentaire" [-f ma-clef]`
   → fichiers `ma-clef` et `ma-clef.pub` dans `/home/moi/.ssh/`

2. Installation de la clé publique
   `ssh-copy-id [-i ma-clef] [user@]distant`
   ou bien `scp` + `slogin` + `cat ... >> authorized_keys`

3. Connexion sans mot de passe
   `slogin [-i ~/.ssh/ma-clef] user@distant`

4. Pour aller plus loin : TP utilisation d'un agent SSH
   - 4.1 Protéger la clé existante par un mot de passe
   - 4.2 Comment ne pas retaper le mot de passe ?
   - 4.3 `ssh-agent` cf gnome-keyring...
   - 4.4 `ssh-add /.ssh/ma-clef` puis `ssh-add -l`

Complément : configuration SSH *110.3*

Exemple de fichier /home/USER/.ssh/config

```
Host eniac
Hostname eniac.moore.upenn.edu.
IdentityFile /home/gallegre/.ssh/eniac_rsa
User gallegre
Port 22

Host hal
Hostname hal9000.nasa.gov.
ServerAliveInterval 30
ServerAliveCountMax 120
```

man 5 ssh_config

Panorama des shells - 1/2

103.1

- Référence
  cf Wikipedia, *Comparison of command shells*

- Shells historiques
  - `sh` original (1971), K. Thompson, Unix AT&T
    mode interactif seulement
  - Bourne shell (`sh`, 1977), Bell Labs, Unix v.7
    ajout des scripts
  - C shell (`csh`, 1978), Bill Joy, Unix BSD
    descendant du Thompson, syntaxe plus proche du C

Panorama des shells - 2/2

103.1

- Shells courants
  - `tcsh` (1981, Tenex C shell), Ken Greer (Carnegie-Melon U.)
    par défaut sur FreeBSD
  - `ksh` (1982), Korn shell, Bell Labs : longtemps propriétaire
    évolutions ksh88 (POSIX), ksh93
  - `bash` (1987) Bourne Again Shell (projet GNU)
    par défaut sur GNU/Linux (GPL) ; v4.0 en février 2009
  - `zsh` (1990), Paul Falstad (Princeton U.)
    probablement le plus riche en fonctionnalités
- Shells restreints
  - `(d)ash`, Kenneth Almquist (sh compact)
  - `sash`, stand-alone shell (commandes internalisées)
- Changer de shell par défaut : `chsh`

Les fonctionnalités du shell *103.1*

- Mode interactif
  - complétion automatique
  - historique des commandes, recherche... (readline)
  - alias
  - ...
- Fonctionnalités mixtes
  - boucles (for, while...)
  - enchaînements de commandes et valeurs de retour
  - fonctions
  - développement (globbing, variables...)
  - fichiers de configuration (`bashrc`...)
  - ...
- Mode script
  - gestion des paramètres ($1, $2...)
  - tests et conditions (if ... then ... else)
  - ...

Documentation *103.1*

- Documentation électronique
  - `man bash`
  - `help help`

- Documentation libre
  - *Advanced Bash Scripting Guide*, M. Cooper (6.0, mars 2009)
    VF : *Guide avancé d'écriture des scripts Bash* (5.3)
  - *Bash Guide for Beginners*, M. Garrels (1.11, déc. 2008)
    VF : *Guide Bash du débutant* (avril 2007)
  - nombreux *tutoriels bash*, plus courts ou plus ciblés

- Livres
  - *Programmation shell sous Unix/Linux*, Ch. Deffaix Rémy, ENI
  - *Introduction aux scripts shell*, A. Robins, N. Beebe, O'Reilly

Complétion

105.1

Complétion standard

- noms de commandes
- entrées de répertoires (fichiers...)

Complétion

105.1

Complétion standard

- noms de commandes
- entrées de répertoires (fichiers...)

Complétion étendue

- `shopt -s progcomp`
- `source /etc/bash_completion`
- sous-commandes
- options longues
- fichiers distants (ssh...)
- ...

Readline - historique

105.1

- `history`
  stockage dans `/.bash_history`
- édition accélérée
  - `C-a`, `C-e`, `C-←`, `C-→` : déplacements
- recherche et parcours de l'historique
  - `man readline` + `/etc/inputrc` : fichier de configuration
- développement de l'historique
  - indicateur d'événement : ex. !!, !123, !#
  - indicateur de mots : ex. 0, 1, ^, $
  - modificateurs : ex. ^chaine1^chaine2^

Rappel : les alias

105.1

Quelques exemples

- `alias ls="ls -color=auto"`
- `alias ll='ls -l'`
- `alias today='date +"%Y%m%d"'`
- `alias rm='rm -I'`

- `alias` seul : liste les alias définis
- `unalias (-a)` détruit un alias défini

Pour aller plus loin : les fonctions

utilisation interactive : “alias à arguments”

Fichiers de configuration *105.1*

- Fichiers principaux
  - `/home/USER/.bash_profile` : shells de login
  - `/home/USER/.bashrc` : autres shells
  - `/etc/profile`
  - `/etc/bash.bashrc`

Fichiers de configuration *105.1*

- Fichiers principaux
  - `/home/USER/.bash_profile` : shells de login
  - `/home/USER/.bashrc` : autres shells
  - `/etc/profile`
  - `/etc/bash.bashrc`

- Contenu
  - Variables d'environnement
    p.ex. prompt : `$PS1, $PS2...`
  - alias
  - fonctions
  - réglages du shell
  - inclusions (`source`)

Configuration du shell *105.1*

- Variables d'environnement
  - `$SHELLOPTS`
  - `$PS1`, `$PS2` ...
  - `$GLOBIGNORE` ...

- `set -f/+f` ou `set -o OPTION`
- `help set`

- `shopt -s / -u` (set / unset)
  - env. 40 options booléennes : `shopt -p`
  - + 27 options "à la set" : `shopt -o -p`

Bash - les "développements" *105.1*

Sept types de développements successifs (*expansions*)

1. développement des accolades { } : factorisation
2. développement du tilde ~ ou ~user
3. développement des paramètres et variables
4. substitution de commande : `` `cmd` `` ou `$(cmd)`
5. développement arithmétique
6. découpage en mots
7. développement des chemins (globbing)

Bash - les "développements" *105.1*

Sept types de développements successifs (*expansions*)

1. développement des accolades { } : factorisation
2. développement du tilde ~ ou ~user
3. développement des paramètres et variables
4. substitution de commande : `` `cmd` `` ou `$(cmd)`
5. développement arithmétique
6. découpage en mots
7. développement des chemins (globbing)

Rappel : les protections

- guillemets doubles
- guillemets simples : plus "forts"
- antislah : un caractère

Diagnostic et enchaînements *105.2*

Valeurs de retour et booléens du shell

- $? : valeur de retour du dernier processus terminé
- 0 = OK $\Longrightarrow$ vrai !
- >0 = erreur $\Longrightarrow$ faux !

Diagnostic et enchaînements *105.2*

Valeurs de retour et booléens du shell

- `$?` : valeur de retour du dernier processus terminé
- 0 = OK $\Longrightarrow$ vrai !
- >0 = erreur $\Longrightarrow$ faux !

Enchaînement des commandes

- ET : `mkdir Toto && cd Toto`
- OU : `mkdir Titi || echo "erreur d'écriture"`
  $\rightarrow$ `mkdir Tutu && echo "OK" || echo "impossible"`
- enchaînement : `cmd1 ; cmd2`
- en parallèle + arrière-plan : `cmd1 & cmd2` && $\neq$ &

Métaprogrammation *103.4*

- La commande `xargs`
  ex. `find /etc/ -size +100k | xargs wc -l`

- La substitution de commande
  ex. `wc -l $( find /etc -size +100k )`
  ou `wc -l ‘find /etc -size +100k‘` (backquotes)
  `echo "Vous êtes connecté sur $(uname -n)."`

- Remarque : la substitution de commande est plus générique (mais plus gourmande).

Redirections étendues : HERE... *105.1*

- HERE-Documents <<

```
$ wall <<FIN
> ETEIGNEZ VOS MACHINES
> coupure electrique imminente
> --- l'equipe systeme
> FIN
```

- HERE-Strings <<<

  ex. `cut -b cut -b 1,3-5,16- <<< "internationalisation"`

Développement des paramètres et variables *105.1*

Évaluation arithmétique

105.1

- Bash standard

```
i=0
i=$i+1      # "0+1"
i=i+1       # "i+1"
```

- Typage numérique (entier)

```
declare -i n=5
n=n+1       # "6"
a=n+1       # "n+1"
```

- Évaluation arithmétique

```
i=0
i=$(( i+1 ))      # standard
(( i=$i+1 ))      # extensions bash...
(( i=i+1 ))
let i=i+1
let i=$i+1
```

Écrire une boucle numérique *105.2*

- La commande `seq`
  - `for i in $(seq 0 2 8) ; do echo $i ; done`
  - `seq 8` 1 à 8
  - `seq 0 8` 0 à 8
  - `seq 0 2 8` 0, 2, 4, 6, 8
- Bash, mode standard

  `while [ $i -lt 9 ] ; do echo $i ; let i=i+1 ; done`
- Bash, mode arithmétique
  1. `while ((i<9)) ; do echo $i ; done`
  2. `for ((i=0; i<9; i+=2)); do echo $i; done`
- Bonus : formatage numérique
  1. `printf 'James Bond %03d, No %02d' 7 3`
  2. `seq -f '%03.0f' 0 2 12` format virgule flottante (!)

Scripts shells

105.2

- Modèles d'exécution
  - exécution `bash monscript.sh`
  - ou exécution avec `# /bin/sh`
  - inclusion : `source script`

Scripts shells

105.2

- Modèles d'exécution
  - exécution `bash monscript.sh`
  - ou exécution avec `# /bin/sh`
  - inclusion : `source script`
- Paramètres positionnels
  - $0, $1, $2...
  - $# nombre d'arguments
  - "$*" la liste des arguments, sans tenir compte des blancs
  - "$@" la liste des arguments, en tenant compte des blancs
  - shift

Un exemple : bonjour.sh

105.2

```
#! /bin/sh

echo "je suis $$"
echo "bonjour $NAME"
NAME="Guillaume"
echo "bonjour $NAME"
exit 0
```

Rappel

$$: numéro du processus courant

Panorama des structures de contrôle *105.2*

- Tests
  - `test` ou `[ ... ]` — test standard
  - `[[ ... ]]` — test avancé (Bash)
- Conditions
  - `if ... then ... fi`
  - `if ... then ... elif ... else ... fi`
  - `case MOT in MOTIF) ... esac`
- Boucles
  - `for VAR in VALEURS ...; do ... done` — énumération
  - `for (( E1; E2; E3 )); do ... done` — numérique
  - `while ...; do ...; done` — tant-que
  - `until ...; do ...; done` — until
  - `select MOT in VALS; do ... done` — menu (boucle interactive)

Exemples de tests

105.2

- Tests standard `[ ... ]`- exemples
- Tests avancés `[[ ... ]]`- exemples

Boucle `for` *105.2*

- Usage interactif (ligne de commande)
  - `for VAR in un deux trois ; do echo $VAR ; done`
  - `for F in *.txt ; do wc -l $F ; done`
- Usage en script

```
for ARG in $@
    do
        echo $ARG
        ...
    done
```

- Variante `select` (en script)

while et until

105.2

La condition if

105.2

- En ligne de commande
  - `if mkdir Rep ; then echo Fait ; else echo Erreur ; fi`
  - cf `mkdir Rep && echo "Fait" || echo "Erreur"`

La comparaison case

105.2

105.2

TP scripts 1 : disable/enable

105.2

1. Ecrire un script `disable.sh` qui
   - prend en argument un nom de fichier
   - le renomme en lui ajoutant le suffixe `.OFF`

2. Ecrire le script inverse, `enable.sh`, qui supprime le suffixe `.OFF`. Il doit accepter en argument les deux variantes `fichier` et `fichier.OFF`.

3. Transformer les deux scripts en un seul `xable.sh`, qui prend une option (-d ou -e) pour indiquer le sens de l'opération.

TP scripts 2 - boucles

105.2

- Avec `find`
  - Exo : Trouver le nb d'entrées de répertoire de chaque type sous `/`, sans changer de système de fichiers (`-xdev`).
  - Exo : Pour les quatre types minoritaires, afficher les entrées.

TP scripts : gestion des liens *105.2*

1. Ecrire un script `rmlink.sh` qui
   - prend en argument une entrée de répertoire
   - la supprime si c'est un lien symbolique
   - retourne un message d'erreur sinon

2. Variante `rmbrlink.sh` : supprime seulement les liens cassés

3. Variante : transforme `rmbrlink.sh` en option (-b) de `rmlink.sh`

4. Ecrire un script `rmhlink.sh` qui supprime l'entrée de répertoire si c'est un fichier régulier avec (ref>1), autrement dit si c'est un lien dur.

5. Ecrire une fonction `ireadlink` qui affiche une résolution de lien symbolique avec intermédiaires : ex. `/usr/bin/rsh -> /etc/alternatives/rsh -> /usr/bin/ssh`.

Les fonctions

105.1

La commande `function`

```
function lprman
    {
    man -t $1 > $1.ps
    lpr $1.ps
    }
```

Tableaux en Bash 1/2 : index numériques *105.2*

- Exemples

```
declare -a Tab

Tab[0]=zero
Tab=(zero un deux trois quatre cinq)

echo ${Tab[2]}
echo ${Tab[*]}
echo ${Tab[*]:2:3}
```

- TP

Tableaux en Bash 2/2 : tableaux associatifs *105.2*

- Tableaux associatifs (depuis Bash 4)

```
declare -A Asso
Asso[couleur]=rouge
Asso=([couleur]=rouge [outil]=marteau [animal]=lion)

declare -p Asso
echo ${Asso[couleur]}
echo ${Asso[*]}
echo ${!Asso[*]}
for KEY in ${!Asso[*]}; do
  echo "$KEY => ${Asso[$KEY]}"; done
```

- TP : Trouver la place occupée par les fichiers de chaque type MIME dans le répertoire utilisateur.
  Astuce : utiliser la commande `file -i` pour les types MIME.
  Variante : remplace le type MIME par l'extension.

sed, expressions rationnelles

Expressions rationnelles (ou régulières) *103.7*

- un “outil” commun à de nombreux utilitaires
  `grep`, `sed`, `awk`, `vim`...
- Deux formes (malheureusement !)
  - forme “basique” interne à chaque commande
  - forme “étendue” standardisée (POSIX.2)
  - `man 7 regex`

sed - Stream EDitor *103.7*

Contexte

- écrit par Lee McMahon en 1973/1974 (Bell Labs),
- dérivé de l'éditeur monoligne `ed`
- applique une série de règles d'édition de texte...
- à chaque ligne d'un fichier, successivement
- reconnaît deux types d'expressions régulières

sed - Stream EDitor *103.7*

Contexte

- écrit par Lee McMahon en 1973/1974 (Bell Labs),
- dérivé de l'éditeur monoligne `ed`
- applique une série de règles d'édition de texte...
- à chaque ligne d'un fichier, successivement
- reconnaît deux types d'expressions régulières

Quelques exemples

- `sed -e "s/Old/New/g" f-in > f-out`
- `sed -e '/^ *$/d' f-in`

awk

AWK - un filtre-langage

- Origines...
  - langage défini par Aho, Weinberger, Kernighan en 1977
  - standard POSIX, NAWK (New AWK), courant 1980s
  - *The AWK Programming Language*, 1988
  - plusieurs interpréteurs libres (orig-awk, gawk, mawk...) ou pas
  - une syntaxe intermédiaire entre C et le shell
  - à l'origine de Perl

AWK - un filtre-langage

- Origines...
  - langage défini par Aho, Weinberger, Kernighan en 1977
  - standard POSIX, NAWK (New AWK), courant 1980s
  - *The AWK Programming Language*, 1988
  - plusieurs interpréteurs libres (orig-awk, gawk, mawk...) ou pas
  - une syntaxe intermédiaire entre C et le shell
  - à l'origine de Perl
- Caractéristiques principales
  - conçu pour analyser un fichier (ou flux) texte divisé en champs
  - tableaux associatifs
  - expressions régulières
  - bien adapté à des scripts unilignes (comme `sed`)

AWK - un filtre-langage

- Origines...
  - langage défini par Aho, Weinberger, Kernighan en 1977
  - standard POSIX, NAWK (New AWK), courant 1980s
  - *The AWK Programming Language*, 1988
  - plusieurs interpréteurs libres (orig-awk, gawk, mawk...) ou pas
  - une syntaxe intermédiaire entre C et le shell
  - à l'origine de Perl
- Caractéristiques principales
  - conçu pour analyser un fichier (ou flux) texte divisé en champs
  - tableaux associatifs
  - expressions régulières
  - bien adapté à des scripts unilignes (comme `sed`)
- Particularités des implémentations

mawk performances et efficacité (précompilé)
gawk richesse et documentation (i18n)
xgawk extensions XML, PostgreSQL
awka comilateur AWK -> C

AWK - invocation et structure

- Invocation de awk
  - `awk -f script.awk fichier`
  - `awk 'code AWK' fichier`
  - exécutable commençant par `#! /bin/awk -f`

AWK - invocation et structure

- Invocation de awk
  - `awk -f script.awk fichier`
  - `awk 'code AWK' fichier`
  - exécutable commençant par `#! /bin/awk -f`

- Structure d'un script

```
motif  { action }
...
```

  - motif : sélecteur de lignes ou BEGIN ou END
  - action : instruction de type procédural

AWK - invocation et structure

- Invocation de awk
  - `awk -f script.awk fichier`
  - `awk 'code AWK' fichier`
  - exécutable commençant par `#! /bin/awk -f`

- Structure d'un script

```
motif  { action }
...
```

  - motif : sélecteur de lignes ou BEGIN ou END
  - action : instruction de type procédural

- Quelques exemples

```
awk 'BEGIN { print "Bonjour !" }'
awk 'length($0) > 60' /etc/passwd
awk 'NR % 2 ==0' /etc/passwd
awk 'BEGIN {FS=":"}  NR % 2==0 {print $1}' /etc/passwd
```

AWK - TP avec find

- Utilisation basique
  Trouver la place occupée par l'ensemble des fichiers de plus de 1 Mo dans le répertoire utilisateur (on peut varier les critères...).

- Utilisation avancée : tableaux associatifs
  Trouver la place occupée par les fichiers de chaque type d'extension (txt, sh, ...) dans le répertoire utilisateur
  Astuce : utiliser la directive `split` pour les extensions.

Gestion des utilisateurs

Comptes utilisateurs *107.1*

Fichiers concernés

- `/etc/passwd` et `/etc/shadow`
- `/etc/group` et `/etc/gshadow`
- `/etc/skel/`
- `/etc/shells`

Commandes

- useradd / userdel (standard, paquet `passwd`)
- adduser / deluser (extension Debian) + `/etc/adduser.conf`
- passwd

Création des comptes :

- manuelle : modification `/etc/passwd`, `/etc/shadow`...
- `adduser john` interactive
- `adduser john ...` en ligne de commande

Anatomie des fichiers de configuration *107.1*

/etc/passwd

1. nom de connexion de l'utilisateur (login)
2. mot de passe chiffré (ou x $\Longrightarrow$ **cf** `shadow`)
3. identifiant numérique de l'utilisateur (UID)
4. identifiant numérique du groupe principal de l'utilisateur (GID)
5. nom complet + commentaires (Gecos)
6. répertoire personnel de l'utilisateur
7. shell de l'utilisateur (ou `/usr/sbin/nologin`)

Compléments

- `man 5 passwd`
- fichiers `adduser.conf` et `deluser.conf` (Debian) : réglages

Entrée /etc/shadow 1/2 *107.1*

Structure du mot de passe

- ex. `allegre:$1$RkDDTG8j$SEpWR3cnmpwjPWAmhwReS1:...`
- 1. login utilisateur
- 2. mot de passe chiffré haché (MD5, SHA1 ...)
  1. 1 = hachage MD5, 6 = SHA-512 (`man 3 crypt`)
  2. Sel : valeur aléatoire différente pour chaque entrée
  3. Mot de passe chiffré (hachage cryptographique)
- 3+. 7 paramètres de validité du mot de passe (à suivre)

Commandes liées

- `mkpasswd` (`whois`)
- `pwgen` (`pwgen`)
- `md5sum, sha1sum, sha256sum...` (`coreutils`)
  - calcul des sommes de contrôle
  - vérification (check)

Entrée /etc/shadow 2/2 *107.1*

Paramètres de validité du mot de passe

1. dernier changement de mot de passe (jours depuis 1970-01-01)
2. âge minimum du mot de passe avant changement
3. âge maximum du mot de passe
4. période d'avertissement (jours avant expiration)
5. période de grâce (*inactive*) (jours après expiration)
6. fin de validité (jours depuis 1970-01-01)
7. réservé

Commandes et fichiers liés

- `chage -l <username>` : paramètres actifs
- `chage [-options] <username>` : modifier les paramètres
- `man 5 shadow`, `man chage`
- module `pam_unix` : application des règles shadow

Gestion des groupes *107.1*

Commandes usuelles

1. `groups <username>` : afficher l'appartenance d'un utilisateur
2. `addgroup <groupe>`
3. `delgroup <groupe>`
4. `adduser <username> <group>`

Pour aller plus loin

- `gpasswd` : administer /etc/group and /etc/gshadow
- définir un mot de passe de groupe
- `newgrp` changer de groupe effectif
- différenciation groupe effectif / groupe principal

NSS (Name Service Switch) *107.1*

- Origine : Sun Microsystems
  - D'abord pour NIS (Network Information Services), ex. YP
  - Puis adapté à LDAP, BDB, ...
- Abstraction des "bases de données" système
  - **utilisateurs** (password + shadow)
  - groupes (groups + gshadow)
  - hôtes (hosts)
  - ...
- En pratique
  - implémenté dans la libc
  - configuration `/etc/nsswitch.conf` (5)
  - commande `getent`(1)
  - développeurs : `getpwent`(3) ...
  - auxiliaire : `nscd`, démon de cache NSS (optionnel)

Supervision des connexions *107.1+*

- Qui est connecté (à l'instant) ?
  - `who (-a)` montrer qui est connecté
  - `w` montrer les utilisateurs connectés et les processus
  - source `/var/run/utmp`
- Qui s'est connecté (dans le passé) ?
  - `last` liste des utilisateurs dernièrement connectés
  - `lastb` liste des tentatives infructueuses
  - `lastlog` dernière connexion de chacun
  - `/var/log/wtmp` (last, écrit par `pam_unix`)
  - `/var/log/btmp` (lastb)
  - `/var/log/lastlog` (écrit par `pam_lastlog`)

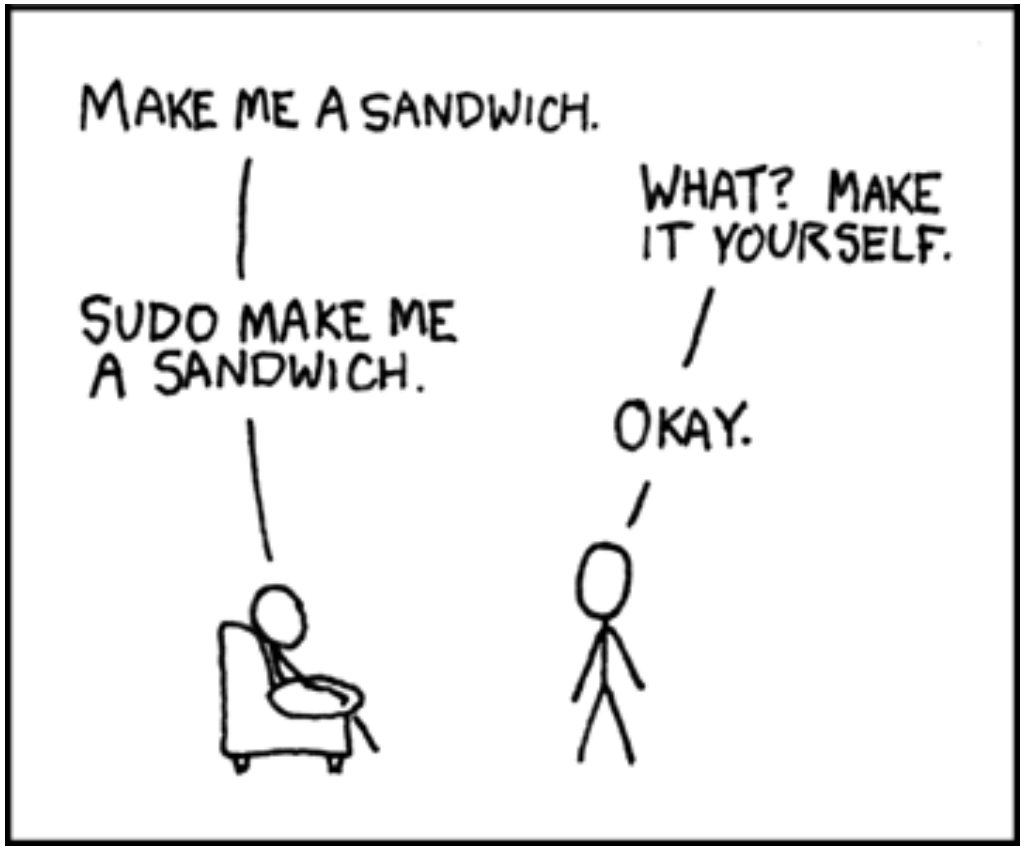
Les sudoers

110.1

- Le fichier de configuration : `/etc/sudoers`
  - des définitions d'alias (4 types)
    - `User_Alias` utilisateur source
    - `Runas_Alias` utilisateur cible
    - `Host_Alias` machine hôte
    - `Cmnd_Alias` commande
  - des autorisations :
    UTILISATEUR HOTE = (EN-TANT-QUE) COMMANDE
    - `root ALL = (ALL) ALL`
    - `%grh ALL = PRINTING, /usr/bin/adduser`

- Les commandes utilisateurs :
  - `sudo (-u <u-cible>) <commande>`
  - `sudoedit <fichier>` ou `sudo -e <fichier>`

XKCD 149 ©Randall Munroe, CC-BY-NC



Exo

1. Accorder à l'utilisateur par défaut les droits de root
2. Autoriser un groupe "secretariat" à créer et supprimer des comptes.

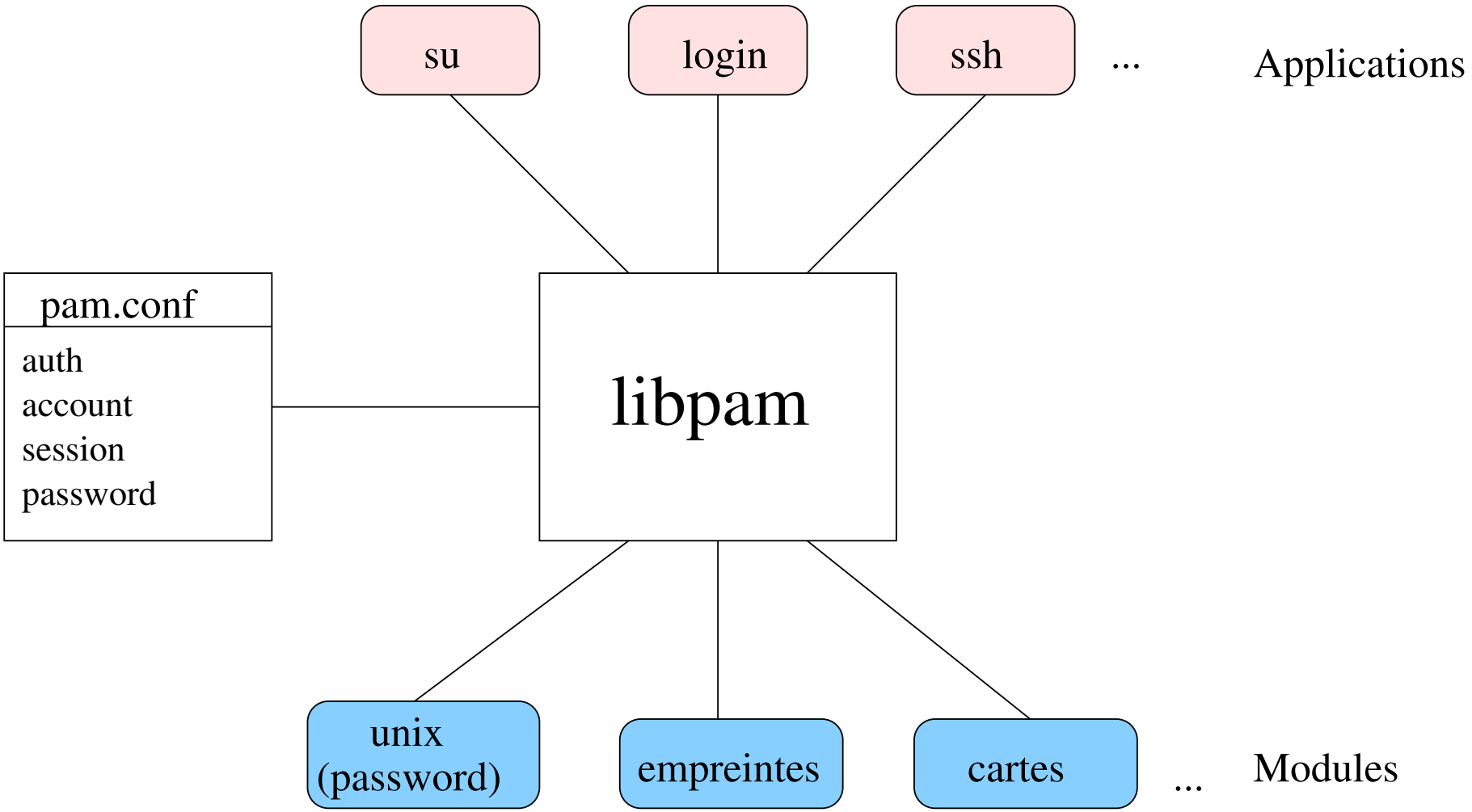
PAM : Pluggable Authentication Modules

- Principe
  - une infrastructure d'authentification unifiée
  - partagée entre le système et les applications
  - un jeu de modules d'authentification
  - extensible et paramétrable par l'administrateur
  - commun à plusieurs Unix : Sun (origine), HP-UX, Linux, FreeBSD

- Paquets de base Debian : `libpam0g` + `libpam-modules`

- Documentation (paquet `libpam-doc`)
  - manpages : pam.conf(5), PAM(7) (extraits du SAG)
  - The Linux-PAM System Administrators' Guide, v1.0
  - The Linux-PAM Module Writers' Guide
  - The Linux-PAM Application Developers' Guide
  - The PAM FAQ

PAM - architecture



PAM - implémentation et services

- Une bibliothèque : `libpam.so` (paquet `libpam0g`)
- Les modules `/lib/security/pam_*.so` (`libpam-modules`)
- Les fichiers de configuration `/etc/pam.d/*` : règles

- Des modules additionnels : paquets `libpam-*`

- Quatre types de services fournis
  - **account** : validité de la connexion
  - **auth**entication : par mot de passe, carte à puce, LDAP...
  - **password** : mise à jour du mot de passe (resp. clé...)
  - **session** : ouverture/fermeture de la session (montage...)

PAM - fichiers de configuration

- Chaque fichier de configuration `/etc/pam.d/service` : règles
- Colonne 1 : type de service (account, auth, password, session)
- Colonne 2 : contrôle : que faire en cas de réussite/échec ?
  - **required** : terminer la pile puis échouer
  - **requisite** : échouer puis retour contrôle à l'application
  - **sufficient** : succès module $\Longrightarrow$ succès final
  - **optional** : important uniquement si le module est seul
  - ou version longue (cf SAG)
- Colonne 3 : module `pam_foobar.so`
- Colonne 4 : arguments du module

- Ex. interdire la réutilisation d'un même mot de passe (option remember=)

TP - prise en main de PAM

1. Créer un utilisateur de test `casimir` et regarder l'effet dans les logs PAM
2. Instrumenter la configuration de `sudo` (par exemple) avec `pam_warn`
3. Faire en sorte que `lastlog` prenne en compte les sessions `su`

4. Interdire l'accès à `casimir` sur `tty2` (`pam_access`)
5. Interdire tous les accès sur `tty2` sauf pour `casimir`
6. Permettre une authentification sans mot de passe à tous sauf `root` sur `tty6`

7. Interdire tous les accès entre 0h et 6h (`pam_time`)
8. ...

XKCD 838

Administration des services

Démarrage de Linux (boot) *102.2*

1. Chargement du BIOS (ou EFI = Extensible Firmware Interface)
2. Gestionnaire de boot (GRUB / LILO)
   - choix du système d'exploitation (et noyau)
   - chargement de Linux avec paramètres noyau
   - programme placé au début du périphérique de boot (MBR)
3. Exécution du noyau
   Diagnostic en console texte
4. init est le premier processus lancé (System V)
   - lit sa configuration dans `/etc/inittab`
   - exécute les scripts d'initialisation de `/etc/init.d/rcS`
   - démarre tous les services du *runlevel* par défaut
5. `getty` en mode console
6. `xdm` / `gdm` / `kdm` (service init.d) (optionnel)

Chargeurs de démarrage (bootloaders) *102.2*

- Principaux chargeurs de démarrage pour PC

  LILO Linux Loader, simple

  GRUB Legacy (0.9x) plus complet, plus complexe

  GRUB 2 réécriture complète, modulaire, complexe

- Fonctionnalités communes
  - capables de chaînage (*chainloader*)
  - interface utilisateur menu ou ligne de commande

- Autres chargeurs

  Das U-Boot (ex-PPCBoot) “universel”

  RedBoot systèmes embarqués

  obsolètes Syslinux (disquettes), Loadlin (DOS)...

LILO (Linux Loader) / ELILO (Efi Lilo) *102.2*

- Inventaire
  - Documentation : manpages `lilo(8)`, `lilo.conf(5)`
  - Commande `lilo` après chaque modification de configuration
  - Fichiers créés (par défaut) : `/boot/map`, `/boot/boot.MMmm`
  - Fichier de configuration `/etc/lilo.conf`

```
boot=/dev/hda
install=menu
prompt
default=Linux

image=/boot/vmlinuz-2.6.26
label="Linux"
root=/dev/hda1
append=""

other=/dev/hda3
label="Windows"
```

GRUB Legacy (v. 0.97)

102.2

- Numérotation “universelle” des disques
  - `(hd0,0)` = `/dev/hda1` (ou `/dev/sda1`)
- Manipulation simplifiée
  - fichier de configuration unique : `/boot/grub/menu.lst`
  - pas de commande à lancer
- Une architecture interne plus complexe : 3 stages

GRUB 2 (v. 1.98)

102.2

- La numérotation a changé
  - `(hd0,1)` = `/dev/hda1` (ou `/dev/sda1`)
  - repérage par UUID ou LABEL conseillé
- Fichiers de configuration
  - Effectif : `/boot/grub/grub.cfg`
  - Reconstruit par `update-grub` ou `grub-mkconfig`
  - Sources multiples :
    - `/etc/default/grub`
    - `/etc/grub.d/*`

De System V aux init basés sur les dépendances - 1 *101.3*

System V et variantes

- SystemV historique
  - `/etc/init.d/*` scripts d'exécution
  - `/etc/rc?.d/*` répartition en *runlevels*
- SystemV init + `insserv` (Debian 6.0 Squeeze)
  - compatible System V init
  - conforme aux dépendances *LSB init*
- Le paquet `file-rc` (obsolète ?)
  - concepts conformes à `sysv-rc`, sans dépendances
  - remplace les liens `rc?.d/*` par un fichier `runlevel.conf`

De System V aux init basés sur les dépendances - 2 *101.3*

Systèmes basés sur les dépendances

- Le système `upstart`
  - initié par Ubuntu (6.10)
  - intègrerait (?) les fonctions de cron, atd, anacron
  - supervise les services lancés
- `systemd`
  - inspiré de `launchd` (MacOS X)
  - Lennart Poettering (RH), *Rethinking PID 1*
  - intégré par Fedora 15 et expérimenté par Debian unstable

init

101.3

init : premier processus

Appelé par le noyau (avec en argument optionnel un run-level / initlevel)

Runlevels

0 extinction

1 *single user* (dépannage, root seulement)

2-5 niveaux utilisateurs

6 redémarrage

S boot (unique)

Les niveaux 2 à 5 sont personnalisables par l'administrateur.

Configuration : `/etc/inittab`

Répertoires associés : `/etc/rc?.d`

Notion de service *101.3*

Trois types de services (environ)

- action : ex. `single`, `halt`, `reboot`...
- configuration : ex. `hdparm`, `ifupdown`, `networking`...
- démon (processus résident) à l'écoute
  - socket unix : mysql, d-bus, acpid...
  - autre IPC (rare)
  - socket réseau : mysql, ssh, cups...

Démons : 2 niveaux de configuration

- applicatif, ex. `/etc/ssh/sshd_config`
- service, ex. `/etc/default/ssh` (Debian) ou `/etc/sysconfig/*` (RH)

Exécution d'un service *101.3*

Lancement

- "haut niveau" : `service ssh start`
- "bas niveau" : `/etc/init.d/ssh start`

Actions normalisées (LSB 4.1 Core, 20.2)

start	
stop	
restart	démarre ou redémarre
try-restart	redémarre le service s'il tourne
reload	relit le fichier de config sans stopper (si possible)
force-reload	relit le fichier de config ou sinon redémarre
status	renvoie l'état (texte + valeur de retour)

Normalisation LSB d'un script init.d - en-tête *101.3*

- Conventions
  - Norme LSB 4.1 Core, 20.3
  - Bloc `BEGIN INIT INFO ...  END INIT INFO`
- Partie gérant les dépendances
  - Provides
  - Required-Start
  - Required-Stop
  - Should-Start
  - Should-Stop
- Partie gérant les runlevels System V
  - Default-Start
  - Default-Stop
- Descriptions...
  - Short-Description
  - Description

SysV-init : un exemple

101.3

Scénario de démarrage sans paramètre noyau

- Linux lance init
- Le run-level n'est pas fixé, donc *initdefault* de `/etc/inittab` $\Longrightarrow$ run-level=2 (Debian...) ou 5 (RedHat...)
- init lance les consoles textes
- Pour chaque lien de type `/etc/rc5.d/K??script`, init arrête le service en lançant `script stop`.
- Pour chaque lien de type `/etc/rc5.d/S??script`, init démarre le service en lançant `script start`.

TP – Manipulation des runlevel *101.3*

1. Vérifier le run-level actuel (`runlevel`)
2. Passer en run-level 2.
3. Lancer le mode graphique manuellement.
4. Tuer le *getty* d'une console. Que constate-t-on ?
5. Repasser en mode de départ. Conclusion ?

Les services : cron

cron : lancement automatisé de tâches

- cron démon (*daemon*) : programme résident en mémoire qui réalise les tâches de fond du système.
- Les tables de tâches (crontab) utilisateurs
- Les tables système ...
- Configuration globale `/etc/default/cron` (Debian)

- Démon `anacron` : services intermittents

Cron utilisateur

- fichier de configuration : `crontab -e`
- Syntaxe : m h dom mon dow command (`man 5 crontab`)
- Permissions : `cron.allow` et `cron.deny` (`man 1 crontab`)
- Spool : `/var/spool/cron/crontabs/`

Exo

1. Ajouter la date dans le fichier `timestamp` toutes les 5 min.

Les crontab système (LSB 4.1 Core, 20.1)

Comment installer un cron "système" ?

1. Utiliser la crontab `root` ou utilisateur dédié $\longrightarrow$ déconseillé
2. infrastructure `/etc/crontab`
   - principal : `/etc/crontab` (+ champ User)
   - auxiliaires : `cron.hourly`, `cron.daily`, `cron.weekly`, `cron.monthly`
3. `/etc/cron.d/*` : format libre

Exemples

- `/etc/cron.daily/find` et `locate`
- `/etc/cron.daily/dlocate` et `dlocate`

Complément : lancement différé

Commande at

- Lancement différé à une date/heure précise
- Exemples
  - `echo "touch /home/stg1/temoin" | at "10:05"`
  - `echo "reboot" | at "17:45 2011-04-30"`
  - `atq` + `at -c <id>`
  - `atrm 3`
- Permissions : `at.allow` et `at.deny` dans `/etc`

Commande batch

Variante : attend une charge système assez basse (< 1.5)

Démon atd

Gère les files `at` et `batch`

Récurrence Très Haute Fréquence ?

- Commande `watch`
  - `watch -n 10 ls -l /var/log/messages`
  - `watch -d ps -F`
  - option `-precise` : un cron THF !

etckeeper : suivi de version sur /etc

- Idée : historique des modifications (issue du développement)
  - une "copie de travail" : `/etc`
  - un référentiel (repository) externe
- Initialisation

```
# aptitude install mercurial etckeeper
# cd /etc
# vim etckeeper/etckeeper.conf -> VCS="hg"
# etckeeper init
# etckeeper commit "import initial"
# hg log -l1
```

- Qu'apporte etckeeper par rapport à Mercurial ?
  - Indication de l'utilisateur "réel"
  - Versionnage des droits (permissions, propriétaires)
  - Nettoyage du référentiel des fichiers "parasites" (`.hgignore`)
  - Prise en compte des installations de paquets (hook apt/yum/...)

TP : etckeeper - prise en main

1. `hg help`
2. modifier un fichier (ex. `/etc/passwd`)
3. `hg status` et `hg diff`
4. `etckeeper commit`
5. `hg log`

6. annuler un changement local : `hg revert`
7. ajouter un utilisateur ; commit atomique
8. `hg blame`
9. annuler un changement commité : `hg revert ...`
10. installer un paquet ; conséquences ?

etckeeper - pour aller plus loin

1. rapatrier sous `/etc` des fichiers extérieurs (ex. GRUB)
2. supprimer du dépôt des fichiers qui changent "sans raison"
3. savoir de quel paquet dépend tel fichier de configuration
4. savoir quels fichiers de configuration ont été déposés par tel paquet

5. examiner les fichiers `/etc/apt.d/*` concernés
6. adapter les scripts automatiques

Les logs

108.2

Tous les événements importants sont consignés dans `/var/log`.

- soit via `syslog` / `rsyslog`
- soit directement par les applications

le service (démon) : syslogd / rsyslog

- collecte les messages de différentes sources
- les analyse (légèrement) et les dispatche

Consultation des logs

- `dmesg` *(noyau : boot + modules)* + `echo 'hello' > /dev/kmsg`
- `last`, `lastlog` *(connexions utilisateurs)*
- `tail (-f)`, `multitail`
- tous les filtres texte : `less`, `grep`...

Évolutions de syslog *108.2*

- `syslog` : un standard BSD, normalisé (RFC 3164)
- Émergence de besoins plus poussés
  - des sources différentes : `syslog`, fichiers ...
  - des backends différents : MySQL, PostgreSQL ...
  - des filtres plus précis : hôtes, calculs, regexps ...
  - des communications sécurisées : fiables, chiffrées
- `syslog-ng` (Balabit, HU)
  - fichier de configuration spécifique
  - définition de modèles : source, destination, log, filtre
- `rsyslog` (Adiscon GmbH, DE)
  - fichier de configuration compatible syslog
  - remplace `syslog` dans Debian depuis Lenny (5.0)
  - architecture modulaire

Le service syslog *108.2*

Composition d'un message

- priorité : 0=debug ... 3=warning ... 5=crit ... 7=emerg
- service (*facility*) (auth mail kern local[0-7] ...)
- texte

Le fichier `(r)syslog.conf`

- sélecteur : <service>.<priorité>
- action : envoi vers
  - fichier, ex. `/var/log/messages`
  - terminal (ou pseudo-term), ex. `/dev/tty8`
  - machine distante (syslog), ex. `@loghost.localdomain`
  - utilisateurs, ex. `root,john` ou tout le monde, `*`
  - pipe **nommé**, ex. `|/var/spool/critMessages`

Client CLI : `logger`

```
logger -p mail.info -t "essailog[$$]" "Bonjour monde"
```

toutes facilities *sauf kernel*

rsyslog - Travaux pratiques *108.2*

Exo

1. Afficher les logs d'authentification sur la console 8.
2. Horodatage de `/var/log/syslog` toutes les 5 minutes.

Exo

1. Passer l'horodatage en format ISO + haute précision
2. Activer la centralisation des logs, en UDP (historique) puis en TCP
3. Ajouter un filtre pour extraire les logs CRON de `auth.log`

Rotation des logs : logrotate *108.2*

- En pratique
  - commande `logrotate` lancée par `cron` (daily)
  - OU forçage manuel `logrotate -f <fichier>`
  - configuration : `/etc/logrotate.conf` et `/etc/logrotate.d/*`
  - état : `/var/lib/logrotate/status`

- Configuration
  - période : `daily, weekly, monthly`
  - OU taille : `size`
  - archivage : `rotate, compress, delaycompress, olddir` ...
  - nommage : `dateext, dateformat` ...
  - scripts : `prerotate, postrotate` et `firstaction, lastaction`

Analyse automatique des logs *108.2*

- `logcheck` (par défaut sous Debian)
  - analyse des logs à intervalles réguliers (1 heure)
  - détection de "traces suspectes"
  - envoi par mail ou vers un fichier, *pipe* ...
  - 3 profils : *paranoid*, *server*, *workstation*
  - 3 niveaux : *system*, *security*, *attack*
- `logwatch` (par défaut sous RedHat)
- pour aller plus loin : IDS (Intrusion Detection Systems)
  OSSEC, Prelude

Analyse interactive des logs *108.2*

- `multitail`
  - suivi de fichiers multiples
  - agrégation de fichiers successifs
  - filtres de recherche et d'affichage

Pour aller plus loin...

- LIRE (LogReport)
  - synthèses et statistiques
  - analyse cross-fichiers
- LogAnalyzer (Adiscon)
  - interface web (PHP)

Administration des Systèmes de fichiers

Diagnostic et vérification d'un système de fichiers *104.2*

- `tune2fs -l` : diagnostic
- `tune2fs -opitions` : optimisation, paramétrage

- `e2fsck` vérification et réparation

- `dumpe2fs` affichage des métadonnées “profondes”

TP - Gestion des systèmes de fichiers 1 *104.1*

Exo 1 : résumé du système de fichiers

1. Trouver le nb d'entrées de répertoire de chaque type sous `/`, sans changer de système de fichiers (`-xdev`).
2. Transformer en script prenant en argument le système de fichiers de départ
3. Pour les quatre types minoritaires, afficher les entrées

Exo 2 : un nouveau montage

1. créer une partition de quelques Go en Ext2fs (avec fdisk...)
2. la rattacher au système de fichiers sur `/mnt/vol`
3. pérenniser ce montage : optionnel, activé par l'utilisateur
4. passer la partition en Ext3 puis en Ext4
5. définir le montage par son label de partition

Commandes : `find`, `fdisk`, `mkfs`, `mount`, `tune2fs`, `e2label`
Fichiers : `/etc/fstab`.

/etc/fstab : montages automatiques *104.3*

- Fichier de configuration `/etc/fstab` : 6 champs
  - Périphérique
    - chemin périphérique, ex. `/dev/sda5`
    - par label, ex. `LABEL=home`
    - par uuid, ex. `UUID=be289e4e-43df-41ba-a3c0-a7366e942e10`
  - Point de montage (répertoire)
  - Type de système de fichiers (ou `auto`)
  - Options de montage (nombreuses)
  - Dump (0, 1) : sauvegardes (quasi-obsolète)
  - Check (0, 1) : priorité de la vérification (fsck) ; 0=aucune
- Options de montage (`man mount`)
  - globales (ex. `ro`, `rw`...)
  - ou spécifiques à un système de fichiers

Identification d'un périphérique *104.3*

1. Périphérique bloc physique
   ex. `/dev/hda1`, `/dev/sda5`
2. Périphérique bloc virtuel
   ex. `/dev/dm-0` ou `/dev/mapper/vg1-lv1` ou `/dev/vg1/lv1`

3. Par label
   - `blkid (-o list)`
   - `findfs LABEL=<monlabel>`
   - `e2label` ou `tune2fs (-l | -L)`
4. Par UUID (similaire)

5. Par liens udev : `/dev/disk/`
   - by-id
   - by-label
   - by-path
   - by-uuid

Périphériques Loopback *104.3*

Exercice 3 : utiliser un CD sans lecteur de CD

1. récupérer l'image ISO d'un CDROM (physique)
   `dd if=/dev/cdrom of=/mnt/cdrom.iso`
2. monter localement l'image dans `/media/image`
   `mount -t iso9660 -o loop=/dev/loop0 /mnt/cdrom.iso /media/image`
3. pérenniser cette configuration, accessible aux utilisateurs

Les loopback : périphériques blocs virtuels

- 8 par défaut : `/dev/loop0` ... `/dev/loop7`
- sinon : `modprobe loop max_loop=8` (ou plus)
- permettent un montage (bloc) d'un fichier image
- `losetup` : fichier $\longleftrightarrow$ périphérique bloc

TP - Gestion des systèmes de fichiers *104.3*

Exo 4 : un nouvel espace de SWAP

1. créer une nouvelle partition de SWAP (avec `parted`)
2. l'activer (`partprobe` si nécessaire)
3. pérenniser cette configuration

Pour aller plus loin : utilisation de Partimage

1. copier quelques répertoires sur la nouvelle partition (exo 2)
2. sauvegarder son image avec `partimage`
3. vandaliser le contenu puis restaurer l'image

Pour aller plus loin avec mount

104.3

Problème posé par atime

Options

- (no)atime
- (no)diratime
- (no)relatime
- (no)strictatime

Types de montage “exotiques”

1. montages multiples
2. montage lié `mount --bind` : système complet ou partiel
3. déplacement `mount --move`
4. partages (miroirs) `mount --make-shared` (multiple)

ELF : Executable and Linkable Format *102.3*

Le format standard des exécutables Linux

- Buts
  - Assembler les unités de compilation (*.o)
  - Créer une image mémoire d'un programme

- Trois sous-types de fichiers ELF

  EXEC binaire exécutable
  REL fichier relocalisable `*.o, *.a`
  DYN fichier objet partagé `*.so`

- Commandes disponibles
  - `file /bin/ls` $\longrightarrow$ ELF 32-bit LSB executable [...]
  - Pour aller plus loin : `readelf -h`, `nm`, `objdump`

Bibliothèques partagées (DYN) *102.3*

- Localisation (rappel) : `/lib` et `/usr/lib` + `/usr/loca/lib`
- Lister les dépendances : `ldd`
  `ldd (-v) /bin/ls`

```
linux-gate.so.1 =>  (0xb78a3000)
/lib/ld-linux.so.2 (0xb78a4000)
libacl.so.1 => /lib/libacl.so.1 (0xb785c000)
...
```

- SONAME : nom canonique de la bibliothèque
  `objdump -p /lib/libacl.so |grep SONAME`
  ex. `ls -l /usr/lib/libasprintf*`

```
/usr/lib/libasprintf.a
/usr/lib/libasprintf.so -> libasprintf.so.0.0.0
/usr/lib/libasprintf.so.0 -> libasprintf.so.0.0.0
/usr/lib/libasprintf.so.0.0.0
```

Bibliothèques partagées : configuration *102.3*

- Fichiers de configuration

  `ld.so.conf` fichier de configuration principal
  `ld.so.conf.d/*` fichiers auxiliaires
  `ld.so.cache` cache (binaire)

- Commandes

  `ldconfig` configuration de l'éditeur de liens dynamique
  `ld.so, ld-linux.so` chargeur et éditeur de liens dynamique

- Variables d'environnement

  `LD_PRELOAD`
  `LD_LIBRARY_PATH`

Astuce : réduire les dépendances

102.3

- Busybox
  - paquet `busybox` : (dépendances sur libm, libc)
  - ou paquet `busybox-static` (autonome)
  - `busybox <commande>`
  - `busybox sh`
  - Usage : dépannage (*rescue*) ou embarqué (*embedded*)

- Autres exemples
  - `dash` : un shell sans dépendances

Pour aller plus loin *102.3*

- Bibliothèques statiques
  - `ar t /usr/lib/libcrypt.a`
  - `readelf -h /usr/lib/libcrypt.a`
  - utile au développeur ou à l'administrateur qui recompile

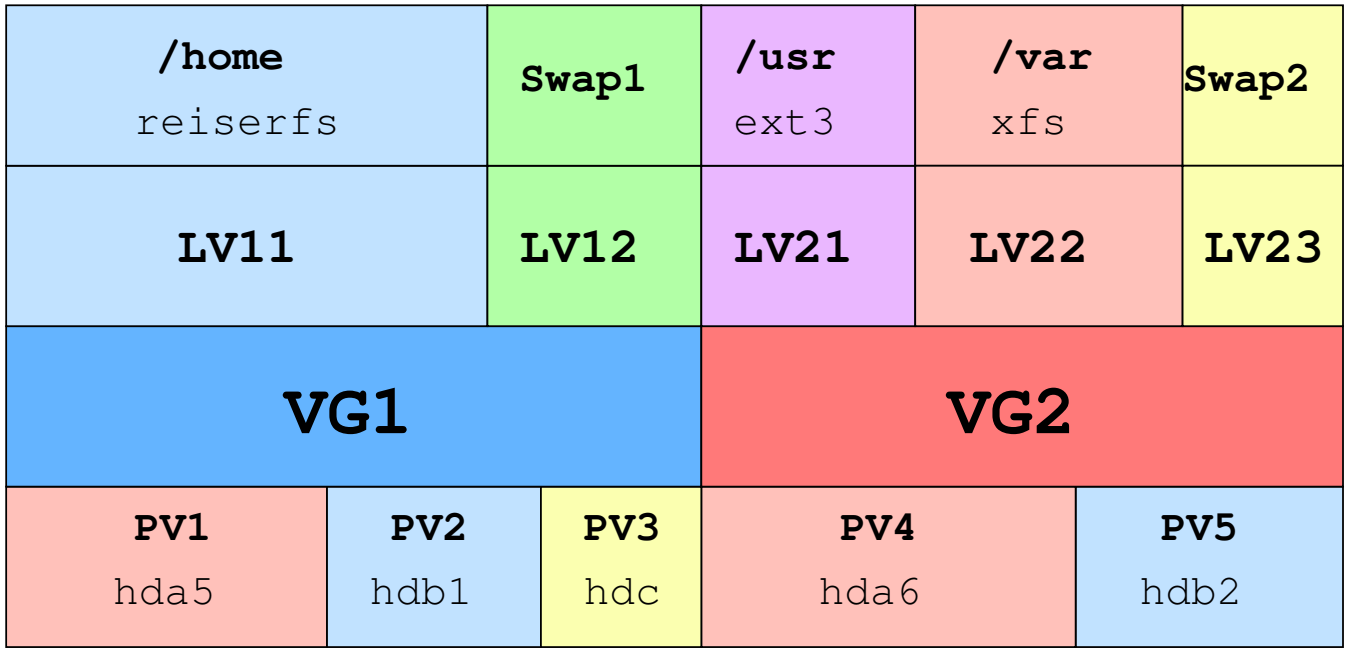
- Explorer un fichier objet ".so"
  `nm -D /usr/lib/libcrypto.so`

Périphériques blocs virtuels

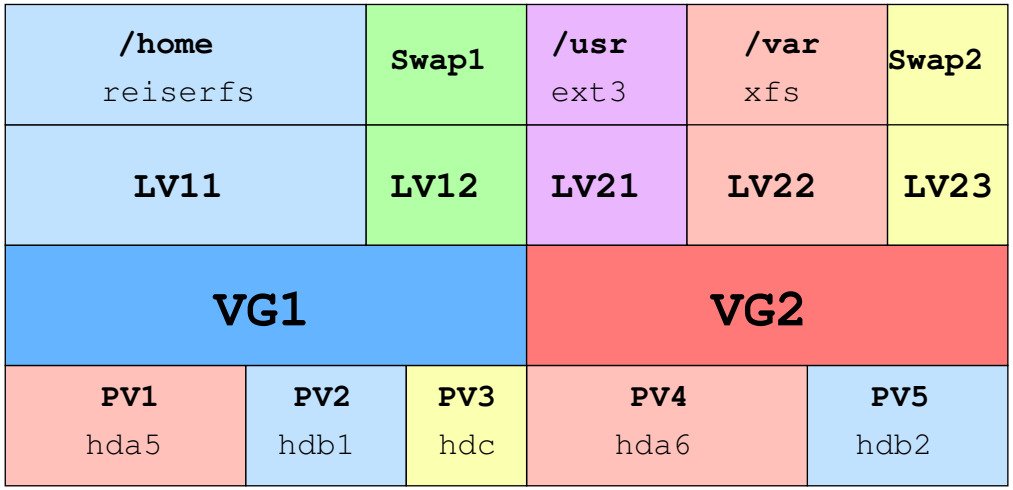
- Device Mapper (Linux 2.6)
  - pilote noyau
  - outils en espace utilisateur : paquet `dmsetup`
- Chiffrement de volume (paquet `dm-crypt`)
- Logical Volume Management (paquet `lvm2`)
  - utilisation plus flexible de l'espace disque
  - IBM AIX (1986-89), HP-UX, Linux 2.2 (1998)
  - Linux 2.6 : lvm2 utilise le Device Mapper
- RAID logiciel (paquet `mdadm`)
  - accès réparti sur plusieurs disques (taille, débit) (RAID lin,0)
  - redondance des données (RAID 1,4,5,6)
- EVMS : un concurrent à LVM+RAID (abandonné)

LVM - les 3 couches

- PV (Physical Volume) : un disque ou une partition
- VG (Volume Group) : un groupe de volumes physiques
- LV (Logical Volume) : un volume logique découpé dans un VG
- La granularité commune
  - PE (Ph. Extent) : une tranche de PV (par défaut 4Mo)
  - LE (Log. Extent) : une tranche de LV (même taille)



LVM - initialisation



- `pvcreate /dev/hda5`
  `pvdisplay (-C)`
- `vgcreate vg-system /dev/hda5 /dev/hdb1 /dev/hdc`
  `vgdisplay (-C)`
- `lvcreate -n lv-users -L 10G vg-system [hda5]`
  `lvdisplay (-C)`
- `ls -l /dev/mapper`

LVM - retailler un système de fichiers

- Nécessite un type de système de fichiers compatible
  - en extension : xfs
  - en extension + réduction : reiserfs, ext3
- Dans un groupe (VG) borné
  - `lvresize`
  - `resize2fs` (ou équivalent)
- En étendant le groupe (VG)
  - `pvcreate /dev/hdb3`
  - `vgextend vg-users /dev/hdb3`
  - terminer comme ci-dessus

LVM - prendre un instantané (snapshot)

- Principe
  - Implémentation du CoW au niveau du périphérique virtuel
  - Unité = Logical Extent (LE)
  - Instantané stocké dans le même VG que l'original

- En pratique
  - `lvcreate -L1G --snapshot --name lv11snap /dev/vg-un/lv11`
  - `lvscan`
  - `lvdisplay /dev/vg-un/lv11`
  - `mount /dev/vg-un/lv11 ...`

- Scénarios d'usage
  - cohérence : instantané "jetable" pendant sauvegarde (BD...)
  - sauvegarde à "faible coût" avant une manipulation risquée
  - ...

LVM - Documentation

- `man lvm` ...
- *LVM Howto*, A.J. Lewis, 2002-2006 (0.19)
  VF : Guide pratique de LVM (0.19-fr)
- *Software RAID Howto*
  LVM+RAID...

Systèmes de fichiers : Unix standard

Une normalisation POSIX

- Inodes : création de liens durs
- Métadonnées standard
  - horodatage : atime, ctime, mtime
  - permissions POSIX
  - propriétaires : utilisateur et groupe
  - type de fichier
- Des IPC standard dans le système de fichiers
  - tubes nommés (pipes)
  - sockets

Des fonctionnalités Unix répandues (ext2 et autres)

- Superblocs
- Fichiers creux (sparse files)

Systèmes de fichiers : caractéristiques avancées - 1

- Journalisation
  - assure la cohérence des données en cas de crash disque
  - ex. VxFS (Veritas, 1991), JFS (IBM), XFS (SGI), ReiserFS, ext3
  - complète ou limitée aux métadonnées
- Métadonnées étendues
  - attributs étendus (xattr) : attribut = valeur
  - listes de contrôle d'accès (ACL POSIX)
- Instantanés...
  - copy on write (CoW) : standard pour la gestion mémoire
  - instantanés : lecture seule
  - clones : écriture aussi (branches)
  - 2 niveaux : périphérique bloc (LVM) ou FS (ZFS, Ext3cow...)
- FUSE : *Filesystem in User Space*

Systèmes de fichiers : caractéristiques avancées - 2

- Allocation
  - Sous-allocation (*tail-packing*...)
  - *Extents*
  - groupes d'allocation (XFS, ZFS)
  - algo. dépendant du périphérique (disque, SSD, hybride...)
- Structure de données
  - BT+ (B-Tree amélioré) : recherche de répertoires
  - H-Tree (B-Tree + hachage) : ext3 (2.6.23+), ext4
- Verrous
- Compression transparente
- Détection et correction d'erreurs

Autres systèmes de fichiers Unix

- JFS (IBM AIX)
  - JFS1 (1990) pour AIX spécifiquement
  - JFS2 (1999) portable, libéré (GPL) en 2000
    JFS2 intégré dans Linux 2.4.19 (juin 2001)
- XFS (Silicon Graphics IRIX)
  - sorti en 1995
  - libéré (GPL) en 2000, intégré à Linux 2.4.23
- ZFS (Sun Solaris, 2005)
- BSD UFS (ou FFS)
  descendant de l'Unix version 7
- BTRFS (Oracle pour Linux, 2007-)
  - développement en cours pour succéder à ext4
  - intégré à Linux 2.6.29 (expérimental)

Voir Wikipedia, *Comparison of Filesystems*

Autres systèmes de fichiers - non Unix

- FAT : File Allocation Table
  - FATnn : 12, 16 ou 32 bits
  - tableau d'allocation ("plan") du disque
  - umsdos / uvfat : métadonnées unix sur FAT (obsolète)
  - usage : compatibilité double boot, clés USB...
- HPFS (IBM OS/2) / NTFS (MS WinNT)
  - High Performance File System (OS/2)
  - New Technology File System (MS WinNT)
- Systèmes à graver
  - Iso9660 : CD, DVD...
  - UDF (Universal Disk Format) : DVD
- Systèmes pour SSD (périphériques flash)
  - techniques de *wear leveling* ou de *log-structured FS*
  - JFFS2, UBIFS
  - LogFS

FUSE : Filesystem in Userspace

- Composants
  - module noyau `fuse` (GPL) depuis 2.6.14
  - `libfuse2` (LGPL) : bibliothèque utilisateur
  - utilitaires `fuse-utils`
  - un paquet par système disponible

- Quelques exemples
  - Réimplémentations : ext2, fat, iso9960
  - Originales : ntfs-3g

  - Réseau montable : fusesmb, fusedav...
  - Réseau non montable : sshfs, curlftpfs, WikipediaFS...

  - Fonctionnalités système : unionfs, mhddfs
  - Versionnage : copyfs
  - Sécurité : clamfs...

Pour aller plus loin : quelques systèmes exotiques

- Fusion de plusieurs systèmes : 3 implémentations
  - UnionFS : noyau
  - AUFS (Another UnionFS) : noyau
  - UnionFSFuse : Fuse
- Systèmes en mémoire
  - tmpfs
  - ramfs (moins évolué)
  - ramdisk (bloc)

Les attributs spécifiques ext2/3/4

Les principaux attributs

i	(immutable) toute modification interdite
a	(append only) accès en écriture sont limités à l'ajout (logs)
A	champ `atime` inchangé (économie, veille)
D	(dirsync) écriture synchrone forcée du répertoire
d	candidat à la sauvegarde par dump
S	(sync) écriture synchrone forcée du fichier
c	compression automatique (non activé)
s	(secure) si effacé, le fichier est d'abord écrasé (non activé)
u	(undel) si effacé, le contenu du fichier est sauvegardé (non activé)

Les commandes

- `lsattr fichiers`
- `chattr [+-=] [AacDdijsSu] fichiers`

Les quotas disque - principe

Ressources concernées

- nombre d'inodes ($\approx$ nb. fichiers)
- nombre de blocs (1 bloc = 4 Ko en général)
- cibles : utilisateurs et groupes

Niveaux de contrainte

- lâche (soft) $\Longrightarrow$ avertissement
- stricte (hard) $\Longrightarrow$ interdiction
- période de sursis
- expiration : contrainte lâche $\rightarrow$ stricte

Les quotas disque - mise en place

Mise en place

1. paquet `quota` et option noyau CONFIG_QUOTA
2. `/etc/fstab` : + options `quota,grpquota`
3. `mount -o remount /dev/hdXN`
4. `quotacheck (-g) -m -c -v /dev/hdXN` $\Longrightarrow$ `aquota.*`
5. `quotaon /dev/hdXN`

Définition des quotas

- `edquota -f /dev/hdXN -u foobar` éditeur (vim...)
- `setquota -u foobar 1000 1500 400 600 /dev/hdXN` blocs (s, h) inodes (s, h)
- `setquota -p u-proto foobar /dev/hdXN` utilisateur “prototype”

Les quotas disque - utilisation

Consultation

- `repquota (-a)` synthèse administrateur
- `quota (-q) (-f /dev/hdXN)` consultation utilisateur

Avertissement

- `warnquota` : envoie un mail à chaque utilisateur contrevenant
- généralement lancé par un `cron` quotidien (distribution)

Les attributs étendus - principe

- Attributs génériques : attribut=valeur
- Espaces de noms des attributs
  - user : accessible à tous
  - trusted : réservés à l'administrateur (userspace)
  - system : réservés au noyau (ex. ACL)
- Recommandations
  - `www.freedesktop.org/wiki/CommonExtendedAttributes`
  - Exemples : `user.mime_type`, `user.charset`, `user.creator`

Les attributs étendus - utilisation

Mise en place

- paquet attr et option noyau CONFIG_EXT2_FS_XATTR=y
- /etc/fstab : + option user_xattr
- mount -o remount /dev/hdXN

Commandes (utilisateur)

- setfattr - -name="user.lang" - -value="fr" fichier
- setfattr -n user.src -v www.april.org fichier
- getfattr -d fichier
- getfattr -m lang -only-value fichier
- man 5 attr

Impact sur d'autres utilitaires

- tar : adoption "en cours" par GNU tar, patchs "distributions", alternative star
- find : sur Solaris (SUN) seulement

Les ACL (Access Control List)

- Norme POSIX 1003.1e
- repose sur les attributs étendus (system)
- permet d'interdire des accès

Six types d'ACL

- USER_OBJ (1) : droits standard du propriétaire
- GROUP_OBJ (1) : droits standard du groupe
- OTHER (1) droits des autres utilisateurs
- USER (0+) utilisateurs supplémentaires
- GROUP (0+) groupes supplémentaires
- MASK (0,1) masque fichier

Algorithme de vérification

1. ACL_USER_OBJ
2. ACL_USER et ACL_MASK
3. (ACL_GROUP ou ACL_GROUP_OBJ) et ACL_MASK
4. ACL_OTHER

Les ACL - mise en place et syntaxe

Mise en place

- paquet `acl`
- option noyau CONFIG_EXT2_FS_POSIX_ACL=y
- `/etc/fstab` : + option `acl`
- `mount -o remount /dev/hdXN`

Syntaxe d'une entrée ACL

`Type:Identifiant:Permission`

1. Type parmi user, group, mask, other
2. Identifiant (Type user ou group) : nom (ex. lisa) ou UID numérique
3. Permission : [rwx]+

Les ACL - utilisation

Exemples d'utilisation

- Accorder un accès lecture-écriture à un utilisateur
  ```
  setfacl -m u:lisa:rw fichier
  ```
- Supprimer tout accès à tout groupe et tout utilisateur via le masque
  ```
  setfacl -m m::rx fichier
  ```
- Supprimer l'entrée correspondant à un groupe
  ```
  setfacl -x g:staff file
  ```
- Dupliquer l'ACL d'un fichier dans un autre
  ```
  getfacl fchier1 | setfacl -set-file=- fichier2
  ```

Documentation

```
man 5 acl
```

Les attributs étendus - principe

- Attributs génériques : attribut=valeur
- Espaces de noms des attributs
  - user : accessible à tous
  - trusted : réservés à l'administrateur (userspace)
  - system : réservés au noyau (ex. ACL)
- Recommandations
  - `www.freedesktop.org/wiki/CommonExtendedAttributes`
  - Exemples : `user.mime_type`, `user.charset`, `user.creator`

Les attributs étendus - utilisation

Mise en place

- paquet attr et option noyau CONFIG_EXT2_FS_XATTR=y
- /etc/fstab : + option user_xattr
- mount -o remount /dev/hdXN

Commandes (utilisateur)

- setfattr - -name="user.lang" - -value="fr" fichier
- setfattr -n user.src -v www.april.org fichier
- getfattr -d fichier
- getfattr -m lang -only-value fichier
- man 5 attr

Impact sur d'autres utilitaires

- tar : adoption "en cours" par GNU tar, patchs "distributions", alternative star
- find : sur Solaris (SUN) seulement

Les ACL (Access Control List)

- Norme POSIX 1003.1e
- repose sur les attributs étendus (system)
- permet d'interdire des accès

Six types d'ACL

- USER_OBJ (1) : droits standard du propriétaire
- GROUP_OBJ (1) : droits standard du groupe
- OTHER (1) droits des autres utilisateurs
- USER (0+) utilisateurs supplémentaires
- GROUP (0+) groupes supplémentaires
- MASK (0,1) masque fichier

Algorithme de vérification

1. ACL_USER_OBJ
2. ACL_USER et ACL_MASK
3. (ACL_GROUP ou ACL_GROUP_OBJ) et ACL_MASK
4. ACL_OTHER

Les ACL - mise en place et syntaxe

Mise en place

- paquet `acl`
- option noyau CONFIG_EXT2_FS_POSIX_ACL=y
- `/etc/fstab` : + option `acl`
- `mount -o remount /dev/hdXN`

Syntaxe d'une entrée ACL

`Type:Identifiant:Permission`

1. Type parmi user, group, mask, other
2. Identifiant (Type user ou group) : nom (ex. lisa) ou UID numérique
3. Permission : [rwx]+

Les ACL - utilisation

Exemples d'utilisation

- Accorder un accès lecture-écriture à un utilisateur
  ```
  setfacl -m u:lisa:rw fichier
  ```
- Supprimer tout accès à tout groupe et tout utilisateur via le masque
  ```
  setfacl -m m::rx fichier
  ```
- Supprimer l'entrée correspondant à un groupe
  ```
  setfacl -x g:staff file
  ```
- Dupliquer l'ACL d'un fichier dans un autre
  ```
  getfacl fchier1 | setfacl -set-file=- fichier2
  ```

Documentation

```
man 5 acl
```

Administration des ressources

Supervision des ressources

- Ressources de type “stock”
  - la mémoire (RAM)
  - la place disque
  - systèmes de fichiers : les inodes
- Ressources de type “flux”
  - le temps processeur : ordonnancement, `nice`
  - les entées/sorties disque : `ionice`
  - la bande passante réseau
- Diagnostic système général
  - `procinfo` : synthèse `/proc`
  - `uptime` : charge et temps d'activité

Supervision de la mémoire *103.5*

- Organisation de la mémoire
  - Mémoire virtuelle = RAM + SWAP
    - Pages de 4 Ko
    - HugePages de 2 à 4 Mo
  - Utilisation par le noyau
    - code
    - cache du système de fichiers
    - structures de données
  - Utilisation par les processus (espace utilisateur)
    - code
    - données : pile + tas

Diagnostic mémoire *103.5*

- `/proc/meminfo` Données brutes

- Mémoire utilisateur
  - `free` Mémoire libre et utilisée du système
    - total = used + free
    - +/- buffers/cache : en vidant les tampons
  - `vmstat` Statistiques détaillées et flux
  - exo : diagnostic mémoire avant et après un `swapoff`

- `slabtop` Caches slab du noyau (experts)

Diagnostic processus et exécutables

- `strace` : tracer les appels systèmes (et les signaux)
  - `strace /bin/ls /`
  - `strace -o ls.strace /bin/ls /` → fichier de sortie
  - `strace -p 1234` → s'attache à un processus lancé
  - `strace -f -o trace -p 1234` → suit également les fils
  - `-e trace=open,close`, `-e trace=file` → filtre les appels
- `ltrace` : tracer les appels de bibliothèques
  - `ltrace -l <bibli>` → limite la trace à cette bibliothèque
  - configuration : `/etc/ltrace.conf`

Exo

1. Trouver les fichiers lus au lancement de la commande `adduser`
2. Vérifier l'activité du serveur de mail local, puis d'un shell actif
3. Mêmes questions pour les appels de bibliothèques

Diagnostic fichiers ouverts *110.1*

- Commandes de diagnostic
  - `fuser` : identifier les processus utilisant un fichier
    - `fuser (-u -v) /dev/audio`
  - `lsof` : idem, et bien plus
    - `lsof /dev/tty1` qui utilise ce fichier ?
    - `lsof -p 1234` quels fichiers sont ouverts par ce processus ?
    - filtres : utilisateur (+u), répertoire (+D), montage (-m)...

- Exercice
  - Trouver les processus qui utilisent les terminaux `tt1` et `tty7`
  - Trouver les fichiers ouverts par le shell courant
  - Trouver tous les fichiers ouverts sous `/home/stg1`

Pour aller plus loin : diagnostic global

- `audit` : strace global
  - `auditd` démon d'audit (avec `auditd.conf`)
  - `auditctl` configurer les règles d'audit
  - `ausearch` recherche dans les logs créés par auditd
  - `aureport` synthèse des logs créés
  - `audispd` multiplexeur d'évènements

- `inotify` : événements sur le système de fichiers
  - Appel système `inotify` depuis Linux 2.6.13
  - Commandes `inotifywatch` et `inotifywait` : paquet `inotify-tools`
  - Dérivées : `incron`, `inosync`, `iwatch`, `gamin`

Sysstat 1/2 : diagnostic à chaud des ressources

Paquet sysstat

- pidstat statistiques sur des tâches individuelles
  - -u (défaut) Usage CPU
  - -d entrées/sorties Disques
  - -r mémoire et fautes de page
  - -w changements de contexte (sWitch)

  ex. `pidstat -d -p 1643 -t 2 5`

- iostat statistiques sur les entrées/sorties

  ex. `iostat -p sda 2 6`

- mpstat statistiques sur les processeurs (mp=multiprocesseurs)

http://sebastien.godard.pagesperso-orange.fr/tutorial.html

Sysstat 2/2 : collecte et analyse de données

Paquets sysstat et isag

sar afficher les mesures de l'activité système

```
sar -u -o datafile 2 3
sar -B -f /var/log/sa/sa29
```

sadf formater les statistiques collectées par sar

```
sadf -d /var/log/sa/sa29 - -B
```

isag visualisation graphique

Fichiers associés dans /var/log/sysstat

sa* fichiers de collecte (binaire), créés par sa1

sar* synthèses quotidiennes (texte), créées par sa2

Localisation et francisation *107.3*

Paramètres régionaux

- choix du clavier
- langue des messages système et des applications
- jeu de caractères
- convention d'affichage (date, monnaie, tri alphabétique...)
- fuseau horaire
- (éventuellement) polices de caractères

Définitions

- **I18N** : (internationalisation) une application est prête à être "traduite"
- **L10N** : (localisation) la traduction est faite pour une langue ou un pays précis

Locales

107.3

Jeux de caractères et locales pour le français

- iso-latin-1 (ou iso-8859-1) : **fr_FR**
- iso-latin-9 (ou iso-8859-15) : **fr_FR@euro**
- UTF-8 : **fr_FR.UTF-8**

Commandes

- `locale -m` : liste des jeux de caractères disponibles
- `locale -a` : locales générées (`/etc/locale.gen`, `/etc/locale.alias`)
- commande `dpkg-reconfigure locales`
- `locale` : variables d'environnement définies et/ou calculées
- `locale -k LC_TIME` : définitions
- `export / unset` LC_ALL / LANG

Autres paramètres régionaux *107.3*

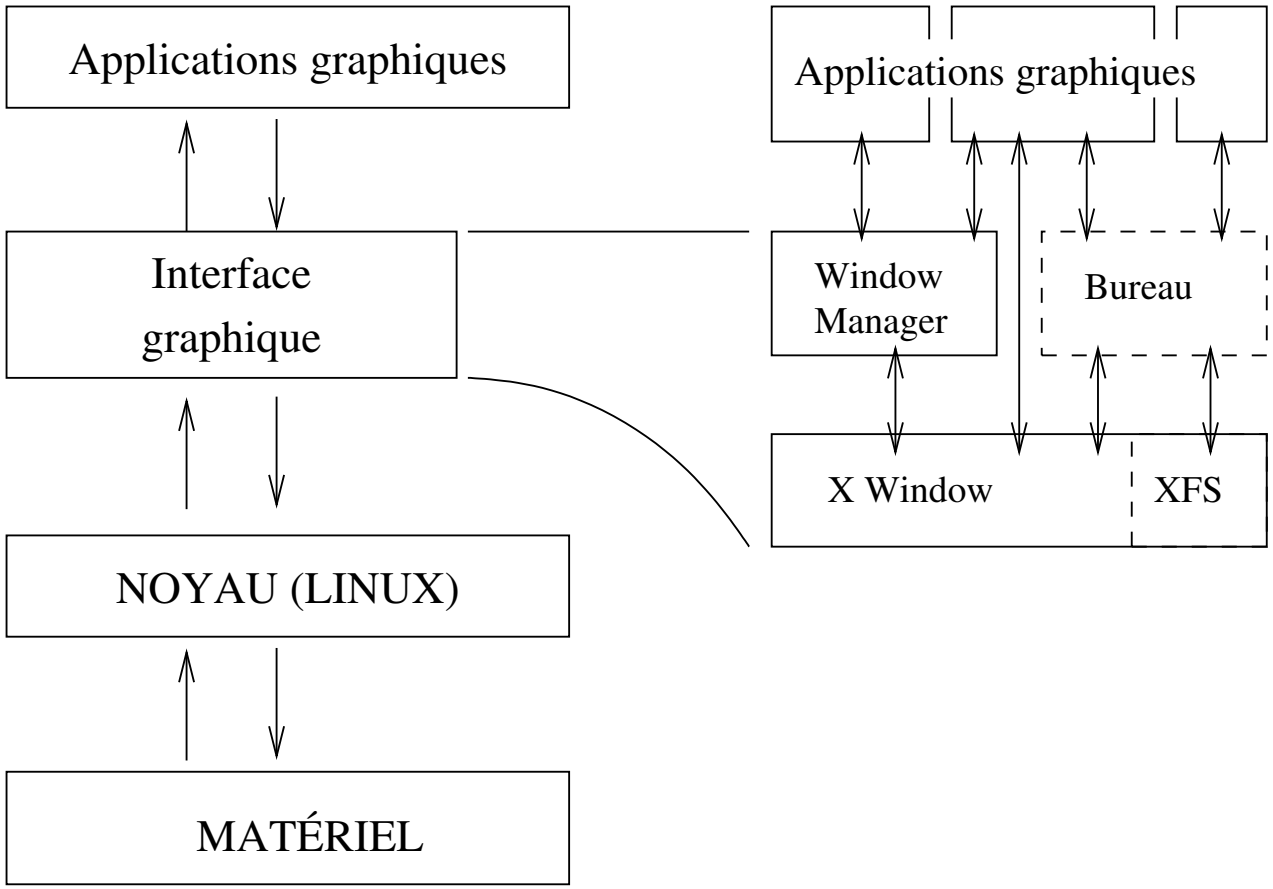
Fuseau horaire

- fichier : `/etc/timezone`
- commandes : `dpkg-reconfigure tzdata`

Configuration du clavier

- multiniveaux : noyau, init (service keymap), udev, X11, bureau...
- fichiers : `/usr/share/keymaps/*`
- commande : `dpkg-reconfigure console-data`

Le système de fenêtrage X-Window (X11) *106.1*



- Système standard sur tous les Unix (sauf Mac OS X)
- Transparence réseau (presque) totale
- Architecture client-serveur !

X Window : historique

106.1

- Historique
  - juin 1984 : X1, MIT
  - jan. 1985 : X6, première version diffusée (propriétaire)
  - sep. 1985 : X9, couleur, licence MIT
  - sep. 1987 : X11, protocole courant
  - mai 1994 : X11R6
  - déc. 2005 : X11R6.9 + X11R7
  - oct. 2009 : X11R7.5
- Implémentations libres
  - XFree86 : 1992 - 2003 (dissolution de l'équipe) - 2008 ...
  - X.org : fork en 2004 (XFree86 4.4rc2), plus dynamique

X11 en pratique

106.1

- Configuration
  - Fichier `/etc/X11/xorg.conf`
  - Optionnel depuis 1.7.0
  - `X -configure` → `xorg.conf.new`
- Lancement
  - Manuel : `/usr/bin/X` pour tester
  - Via `xdm`... (service) en temps normal
- Logs
  - `/var/log/X.?.log`

X11 : principales composantes *106.1*

- Serveur X (`/usr/bin/X`)
- Gestionnaire de session X (X Display Manager)
  ex. xdm, kdm, gdm, slim...
- Bureau graphique (optionnel)
  ex. Gnome, KDE, XFCE...
- Gestionnaire de fenêtres (Window manager)
  ex. metacity, kwm, xfwm4, twm, awesome...
- Console / émulateur de terminal
  ex. xterm, mlterm, xfce4-terminal...

X.org : un système très modulaire *106.1*

- Diagnostic
  - Répertoire `/usr/lib/xorg/modules`
  - Commande `xdpyinfo`

- Exemples
  - Pilotes de cartes video (`drivers`)
    - Nvidia : `nv, nvidia, nouveau`
    - `intel`
    - `ati`
  - Pilotes de périphériques d'entrée (`input`)
    - standard : `kbd`, `mouse`
    - `synaptics`
    - `wacom`
  - Extensions
    - `libdri` : Direct Rendering Infrastructure...
    - `libglx` : MesaGL / OpenGL pour X...

Concepts et commandes X11 *106.1*

- Évènements X11 (clavier, souris, logiciel)
  `xev` : tester les entrées
- Propriétés et informations
  - Commande `xwininfo`
  - Commande `xprop`
- Ressources X
  - Commande `xrdb (-query -all)`
  - Fichiers `/.Xdefaults` et `/etc/X11/Xresources/*`
- Contrôle des fenêtres
  Commande `xkill`

Administration des périphériques et des modules

Modules noyau *101.1*

Paquet : `module-init-tools`

Listing des modules

- modules chargés : `lsmod`
- modules disponibles : `modprobe -l` → `/lib/modules/`
- détails : `modinfo <module>`

Chargement, déchargement

- `insmod`, `rmmod` (obsolètes)
- `modprobe <module> <params>`
- `modprobe -r <module>`
- logs noyau : `dmesg` ou `/var/log/kern.log`

Modules - dépendances et configuration *101.1*

Gestion des dépendances

- `depmod` : calcule les dépendances
- génère `modules.dep(.bin)` et `modules.symbols(.bin)`
- extrait les alias vendor-product : `modules.alias(.bin)`

Fichiers de configuration

- `/etc/modprobe.d/`
  - `aliases.conf`
  - ...
- `/etc/modules` : chargés au démarrage par `/etc/init.d/module-init-tools` (Debian)

Documentations obsolètes

- paquet modutils (2.4), démons kerneld (2.0), kmod (2.2)

Gestion des périphériques - pilotes *101.1*

Point de vue des pilotes système : /dev

- Périphériques blocs
  - disques dur (IDE /dev/hdX, SCSI /dev/sdX...)
  - mémoires flash, SSD, clés USB (/dev/sdX)
  - lecteurs/graveurs CD/DVD (IDE ou SCSI)
- Périphériques caractères
  - interfaces série
  - interfaces parallèle...
  - bus USB, Firewire...
- En commun : identifiant (majeur, mineur)
- Interfaces réseau : PAS des périphériques au sens noyau

Documentation détaillée sur les périphériques

- sources noyau, Documentation/devices.txt
- ou http://www.lanana.org/docs/device-list/

Gestion des périphériques - matériel *101.1*

Point de vue matériel : interfaces de connexion

- Périphériques fixes
  - intégrés à la carte mère : bus PCI, AGP...
  - slots PCI, AGP...
- Périphériques "hotplug"
  - cartes PCMCIA / PCCARD
  - bus USB
  - bus Firewire (IEEE 1394)
  - bus SATA + connecteurs eSATA (externes)

Diagnostic matériel *101.1*

- Examen des bus matériels
  - `lspci` : afficher les périphériques PCI (paquet `pciutils`)
  - `lsusb` : afficher les périphériques USB (paquet `usbutils`)
  - `scsiinfo` : afficher les périphériques SCSI (paquet `scsitools`)
  - `lshw` + `lshw-gtk` : sonder tout le matériel
  - `dmidecode` : afficher les infos DMI / SMBIOS
- Disques durs
  - `hdparm` : configurer / tester les disques IDE et SAS
  - `smartctl` + `smartd` : tests SMART (paquet `smartmontools`)

Terminaux et pseudo-terminaux *101.1*

- Consoles virtuelles (TTY)
  - consoles texte standard (Alt + F1-F8...)
  - `/dev/tty0-63` (4, 0-63)
  - `/dev/tty0` : console virtuelle courante (1 à 6 généralement)
- Ports série
  - terminaux série ou émulation logicielle (+ NULL-modem)
  - `/dev/ttyS0-S3...` (4, 64-255)
- Pseudo-terminaux (PTYs)
  - terminaux X, session shell...
  - `/dev/pts/0...` + `/dev/ptmx`(System V)
  - obsolètes : `/dev/ptyXN`, `/dev/ttyXN` (BSD)
- Compléments
  - `/dev/tty` : console courante (toutes catégories)
  - `/dev/console` : console de log (noyau)
  - cf `Documentation/devices.txt`, section *Terminal devices*

udev : un système /dev dynamique

- Principales caractéristiques
  - n'affiche que les périphériques vraiment présents
  - détecte le branchement à chaud de périphériques et informe les applications utilisateur (via D-BUS et HAL)
  - peut fixer un nom spécifique pour chaque périphérique (ex. `/dev/cleUsb1G` au lieu de `/dev/sde`)
  - peut créer au vol les identifiants majeur/mineur
  - peut charger le pilote (module noyau) et le firmware si nécessaire
  - peut affecter des permissions prédéfinies au périphérique
  - peut lancer des scripts d'initialisation/configuration
  - entièrement géré en espace utilisateur (Userspace /dev)

- Autres composants liés
  - SysFS : vue sur les structures de données du noyau
  - D-Bus (Desktop Bus)
  - HAL (Hardware Abstraction Layer) (obsolète ?)

Udev - composantes

- En espace noyau
  - uevents : événements envoyés par le noyau (via netlink)
  - sysfs : description du périphérique matériel
- Démon `udevd` : écoute les uevents et les passe à udev
- Fichiers de règles
  - `/etc/udev/rules.d/` : modifiables
  - `/lib/udev/rules.d` : distribution
- utilitaire `udevadm`
  - `udevadm info --query=all` : interroge la base de données
  - `udevadm info --attribute-walk` : interroge SysFS
  - `udevadm monitor` : écoute les événements uevents/udev
  - `udevadm test` : teste une règle
  - `udevadm control` : contrôle le comportement du démon
  -

Udev - règles /etc/udev/rules.d/*.rules

Deux types de règles

1. règles de sélection : opérateurs ==, !=
2. règles d'action / affectation : opérateurs =, +=, :=

ACTION== DEVPATH==	
KERNEL(S)== SUBSYSTEM(S)== DRIVER(S)== ATTR(S){...}== ...S==	nom noyau d'après sysfs ascendants
TEST{...}== PROGRAM== RESULT==	fichier existe ? test (code) test (stdout)

NAME= SYMLINK+=	nom alternatif
OWNER= GROUP= MODE=	permissions permissions permissions
RUN= WAIT_FOR= OPTIONS= IMPORT=	exécutable fichier environnement

Documentation : *Writing udev rules*, Daniel Drake

Udev - TP

1. définir une règle udev pour donner un lien fixe à une clé USB
2. la monter automatiquement à l'insertion, sans HAL
3. en passant par HAL + `ivman`

Le noyau Linux

Historique

- Version 0.01 (sept 1991) ... 0.12 (jan 1992)
- Version 0.95 (GPL) ...0.99.15j (mars 1994)
- Version 1.0 : mars 1994 (i386 uniprocesseur)
- Version 1.2 : mars 1995 (+ Alpha, Sparc, MIPS...)
- Version 2.0 : juin 1996 (SMP...)
- Version 2.2 : jan 1999 (1,8 M lignes)
- Version 2.4.0 : jan 2001 (3,4 M lignes)
- Version 2.6.0 : déc. 2003 (5,9 M lignes)

Numérotation : actuelle 2.6.24.4 (A.B.C.D)

- A : version du noyau
- B : révision majeure
- C : révision mineure (nouvelle fonctionnalité, nouveau driver)
- D : correction (bugfix, patch sécurité) : depuis 2.6.11.0
- -XX : branche (-ac, -mm) ou -pre (preversion), -rc (release

Linux - la communauté

Principaux contributeurs

- individuels
- Intel, RedHat, IBM, Novell (Suse), Linux Foundation

Principaux sites

- http ://www.kernel.org : le dépôt principal
- http ://lwn.net/Kernel : Linux Weekly News, articles d'actualité
- http ://kerneltrap.org : d'autres articles réguliers
- http ://kernelnewbies.org : documentation pour "débutants"

Voir *2.6.24 - some statistics* à `http://lwn.net/Articles/263717/`.

Noyau - les paquets Debian

Les paquets

- linux-source-... : $\simeq$ 6
- linux-image-... : $\simeq$ 43
- linux-headers-... : $\simeq$ 41
- linux-tree-... : $\simeq$ 40 paquets virtuels
- linux-modules-... : $\simeq$ 18
- linux-patch-... : $\simeq$ 16
- linux-doc-... : $\simeq$ 5
- linux-manual-... : $\simeq$ 5

- Référence : *Debian Linux Kernel Handbook*
- obsolètres : kernel-source-...

Installation image

- `aptitude install linux-image...`
- Debian configure les chargeurs de démarrage

Recompilation noyau - préparation

- Visite du site www.kernel.org
- Récupérer les sources d'un noyau (par ex. 2.6.24.3)
- dans `/usr/src` pour le groupe "src"
- Préparation et compilation
  - `make help`
  - `make allmodconfig` ou `make menuconfig`...
- Diagnostic matériel
  - `lspci (-v...)` (paquet `pciutils`)
  - `lshw` et `lshw-gtk` (paquets éponymes)

Recompilation noyau - configuration

- `make menuconfig` ou `make xconfig` ou `make gconfig`...
  - 3 états : désactivé, activé, module
  - optimisation compilation (désactivation) et performance
  - bonne connaissance matériel (lspci, lshw...)

- Quelques exemples de paramètres :
  - General Setup > Local : suffixe au n° de version ("-joe")
  - Networking > Networking options > *
  - File systems > *
    - **File systems > Ext2**
    - **File systems > Ext3**
  - Kernel hacking > Magic SysRq key
  - Le plus complexe : Device drivers > *
    - **ATA/ATAPI/... > IDE/ATA-2 DISK support**
    - **ATA/ATAPI/... > ??? chipset support**

Recompilation noyau - compilation + installation

- Compilation
  - `make bzImage modules`
  - `make modules_install install`
- Installation
  - `make install` : copie le noyau dans /boot
  - configuration de LILO / GRUB
  - conserver l'ancien noyau !
  - conserver le `.config`
  - `System.map` pour le débogage du noyau (Kernel Panic)
  - `initrd` pour le chargement des modules

Patch et mise à jour

Patch de mise à jour

- récupération d'une archive patch sur kernel.org
- `patch -p1 --dry-run`
- `make oldconfig` : nouveaux paramètres uniquement
- `make` : compilation partielle
- installation : comme précédemment

Patch de fonctionnalité

- Patchsets officiels : http ://wiki.archlinux.org/index.php/Kernel_Patches_and_Patchsets
- Fonctionnalités supplémentaires, ex. TuxOnIce, etc.

Debianisation

- par le noyau, légère : make deb-pkg
- officielle Debian, pour les modules : `module-assistant`
- officielle Debian, pour le noyau : `make-kpkg`

Administration réseau

Architecture TCP/IP

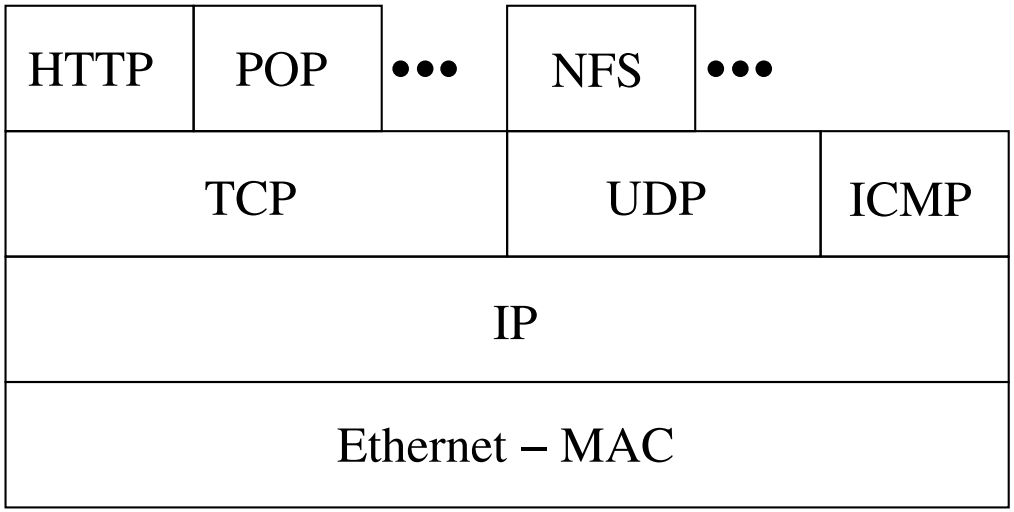
109.1

Un modèle par couches

ernet réseau local Ethernet-MAC

IP l'adressage Internet

TCP le transport



Architecture TCP/IP

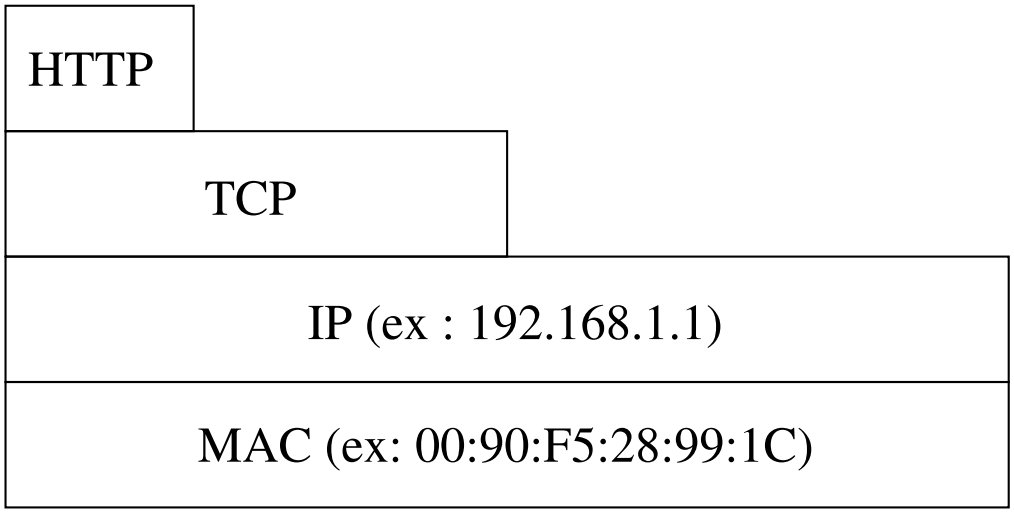
109.1

Un modèle par couches

ernet réseau local Ethernet-MAC

IP l'adressage Internet

TCP le transport



TCP / UDP

109.1

TCP (Transport Control Protocol)

- orienté *connexion*
  paquets ordonnés, type conversation (*stream*)
- fiabilisé : contrôle & correction d'erreur
- plutôt lent
- le plus utilisé par les services usuels

UDP (User Datagram Protocol)

- paquets indépendants
- plus réactif et rapide
- utilisé par NFS et Netbios (SMB)

Premières commandes *109.2*

Commandes

- `ifconfig (-s)`
- ou `netstat -i (-e)`
- ou `ip link (list)`
- ou `ip address (list)`

interfaces

- lo *(interface virtuelle boucle locale)*
- eth0 *(première interface ethernet)*

- adresse MAC : 6 octets ex. HWaddr : 00 :90 :F5 :28 :99 :1C
  Propre à la carte réseau
- adresse IP : déterminée par la topologie du réseau
  - IPv4 : 4 octets, 32 bits ex. inet addr : 192.168.1.1
  - IPv6 : 128 bits *2001 :0db8 :3c4d :0015 :0000 :0000 :abcd :ef12*

Premiers tests *109.2*

ping, ping6

Tester soi-même, un voisin, un absent, le réseau...

Options utiles

- `ping -c5 192.168.1.1` — count=5
- `ping -b 192.168.1.255` — broadcast (souvent désactivé)
- `ping -f -i 0.2 192.168.1.1` — flood + interval

Exo

1. Changer son adresse IP et retester les pings. Conclusion ?
   ```
   ifconfig eth0 192.168.1.100
   ifconfig eth0 192.168.100.1
   ```

Astuce pour simuler un ping broadcast :

```
nmap -sP 192.168.1.15/24
```

Routage, réseau et sous-réseaux *109.2*

Cheminement d'un message

- Un paquet IP est une partie de message TCP (ou UDP, etc.)
- Dans chaque paquet, 2 adresses IP : source et destination

Anatomie d'une adresse IPv4

- $\underbrace{192.168.0}_{\text{réseau}}.\underbrace{1}_{\text{hôte}}$ (classe C) ⟵ réseau local
- $\underbrace{172.116}_{\text{réseau}}.\underbrace{0.10}_{\text{hôte}}$ (classe B)

Adresse, masque de réseau, broadcast.

Notation CIDR (Classless Inter Domain Routing)

192.168.0.1/24 ⟶ 24 bits réseau + 8 bits hôte

`ipcalc` : la calculatrice réseaux

Routage : en pratique *109.2*

Table de routage

Décrit les chemins possibles.

`route (-n)` ou `netstat -r(n)` ou `ip route (list)`

- réseau local
- adresse par défaut (destination 0.0.0.0)

La passerelle (*Gateway, Gw*)

Pour sortir du réseau local, la passerelle interconnecte des réseaux.

Souvent X.Y.Z.254

Modifier le routage

109.2

`route del default`

Quel impact ?

`route add default gw <ip>` où <ip> est l'ip de la passerelle

Revient à la situation initiale

Les routeurs :

Machines spécialisées avec tables de routage complexes

Suivre une route (TTL)

`traceroute (-I|-T|) 91.121.14.67`

`mtr (-t|-g) 91.121.14.67`

Configuration réseau - Debian *109.2*

Rappel configuration manuelle

```
ifconfig eth0 172.16.0.111
  netmask 255.255.255.0 broadcast 172.16.0.255
route add default gateway 172.16.0.1
```

Configuration Debian

Dans `/etc/network/interfaces` :

```
iface eth0 inet static
    address 192.168.0.11
    netmask 255.255.255.0
    broadcast 192.168.0.255
    gateway 192.168.0.254
```

`ifdown eth0 && ifup eth0` ou

`service networking restart`

`man 5 interfaces`

Rappel configuration réseau - RedHat *109.2*

Configuration manuelle

```
ifconfig eth0 172.16.0.111
  netmask 255.255.255.0 broadcast 172.16.0.255
route add default gateway 172.16.0.1
```

Configuration RedHat

Dans /etc/sysconfig/network-scripts/ifcfg-eth0 :

```
DEVICE=eth0
ONBOOT=yes
BOOTPROTO=none
NETMASK=255.255.255.0
GATEWAY=172.16.0.1
TYPE=Ethernet
IPADDR=172.16.0.111
```

`ifdown eth0 && ifup eth0` ou

`service networking restart`

La commande ip

109.2

- la configuration “nouvelle génération” : `ip ss-cmde`
- paquet `iproute`

- ip link : équivalent à ifconfig
- ip address : équivalent à ifconfig
- ip route : équivalent à route

- sous-commandes avancées : multicast, tunnels...

IPv6 : une introduction

Des adresses 128 bits (vs. 32 bits pour IPv4)

- Avantages réels
  - plus de pénurie d'adresses à gérer
  - plus de NAT obligatoire
  - autoconfiguration simplifiée
- Avantages supposés
  - qualité de service (QoS) intégrée
  - connexions sécurisées (IPSec) intégrées
  - routage plus efficace et simplifié
- Contraintes
  - Coexistence IPv4 - IPv6
  - Changements d'habitude !
- Référence : Linux IPv6 Howto, Peter Bieringer

Anatomie d'une adresse IPv6

- Exemple : 2001 :0db8 :3c4d :0015 :0000 :0000 :abcd :ef12
- Notation
  - hexadécimal + deux-points (vs. décimal + point)
  - 128 bits = 16 octets
  - = 32 h-chiffres = 8 quads
  - raccourci : 2001 :db8 :3c4d :15 **:** **:**abcd :ef12
- Composition
  - réseau : 64 bits
  - interface (hôte) : 64 bits
  - $\underbrace{2001 : 0db8 : 3c4d}_{\text{préfixe global}} : \underbrace{0015}_{\text{sous-réseau}} : \underbrace{0000 : 0000 : \mathit{abcd} : \mathit{ef}12}_{\text{interface}}$

Types et intervalles d'adresses IPv6

Préfixe IPv6	Allocation
0000 : :/8	réservé IETF
2000 : :/3	**Unicast global**
FC00 : :/7	Unicast local unique
FE80 : :/10	**Unicast lien-local**
FEC0 : :/10	Unicast site local (obsolète)
FF00 : :/8	Multicast

- Exemples :
  - `2xxx:...`, `3xxx:...` : unicast global
  - `FE8x:...`, `FE9x:...`, `FEAx:...`, `FEBx:...` : lien-local
- Cas particulier
  - localhost : : :1/128

En pratique : premiers tests

- Support d'IPv6 par le noyau Linux ?
  `cat /proc/net/if_inet6` ⟶ interfaces

- Interfaces réseau
  - `ifconfig`
  - inet + inet6 : double pile IP
  - scope = lien-local
  - `ip (-4 | -6 |) addr show`
  - IPv6 dérivée de l'adresse MAC (RFC 4862)
  - ex. `00:19:66:e9:03:81` ⟶ `fe80::219:66ff:fee9:0381`

- `ipv6calc -showinfo (-m) <addrIPv6>`

En pratique : ping6

- La machine locale
  - `ping6 ::1`
  - `ping6 -I eth0 fe80::219:66ff:fee9:381` hôte local
  - attention : lien-local ⇒ préciser l'interface
- Les autres machines
  - `ping6 -I eth0 ff02::1` (ou ip6-allnodes) multicast
  - `ping6 -I eth0 fe80::16da:e9ff:fe76:7b40` autre machine
- Configuration
  - vérifier `/etc/hosts`
  - .

Travaux Pratiques : SSH en IPv6

- `netstat (-4 | -6 |) -ltpn`
- Configuration sshd : `/etc/ssh/sshd_config`
  - `ListenAddress`
  - `AddressFamily`
- Connexion
  - `ssh -l user fe80::219:66ff:fee9:381%eth0`

Résolution de noms (DNS) *109.4*

/etc/hosts

Établit des correspondances *nom d'hôte* $\leftrightarrows$ *adresse IP*

Domaine Name Server (DNS)

- Permet une équivalence entre nom et adresse IP
  - ex. cressida.silecs.info $\leftrightarrows$ 82.67.62.169
  - ex. www.silecs.info $\rightarrow$ silecs.info $\leftrightarrows$ 213.186.33.2 (alias)
  - ex. lear.silecs.info $\rightarrow$ 88.172.133.112 $\rightarrow$ . . . .proxad.net
- Fonctionnement par arborescence de serveurs
  - Dans chaque serveur : cache pour minimiser les requêtes
  - Un *authoritative server* fait autorité pour un domaine

Exemples de TLD

- générique : .com .org .net .name . . .
- pays : .fr .uk .tv .uk .us .eu . . .
- *sponsored : .edu .gov .int .museum .xxx . . .*

Fonctionnement du DNS *109.4*

Modèle client-serveur

- Côté serveur
  BIND 9 majoritaire (Internet Software Consortium)
  Challengers : PowerDNS, Unbound, MS_DNS

- Côté client
  - Bibliothèque partagée *resolver* dans la *glibc*
  - Configuration via `/etc/resolv.conf`
    - serveurs à interroger (nameserver)
    - domaine de recherche par défaut (search)
  - Configuration des priorités
    - `/etc/hosts` est prioritaire sur DNS par défaut.
    - Pour affiner les priorités : `/etc/nsswitch.conf`

Clients DNS

109.4

- Client léger : `nslookup`
- Clients complets :
  - `dig` (dnsutils)
  - `host` (host)
- DNS et IPv6 ?
  - `host (-t A | -t AAAA | ) www.go6.net`
- Sans oublier...
  `ping` (/etc/hosts puis DNS)

DHCP

Obtenir automatiquement les paramètres réseau

DHCP : client/serveur pour

- adresse IP
- routage (passerelle)
- DNS (facultatif)
- WINS, BOOTP, ...

Le parc d'adresses est limité $\Longrightarrow$ *lease* (bail) temporaire

Côté client

`dhclient [interface]` ou `pump -i eth0`

`dhclient -r` : abandon du bail

Côté serveur

- Contrôle des attributions
  - lier une certaine IP à une adresse MAC
  - autoriser uniquement certaines adresses MAC

WHOIS - annuaire des adresses et domaines internet

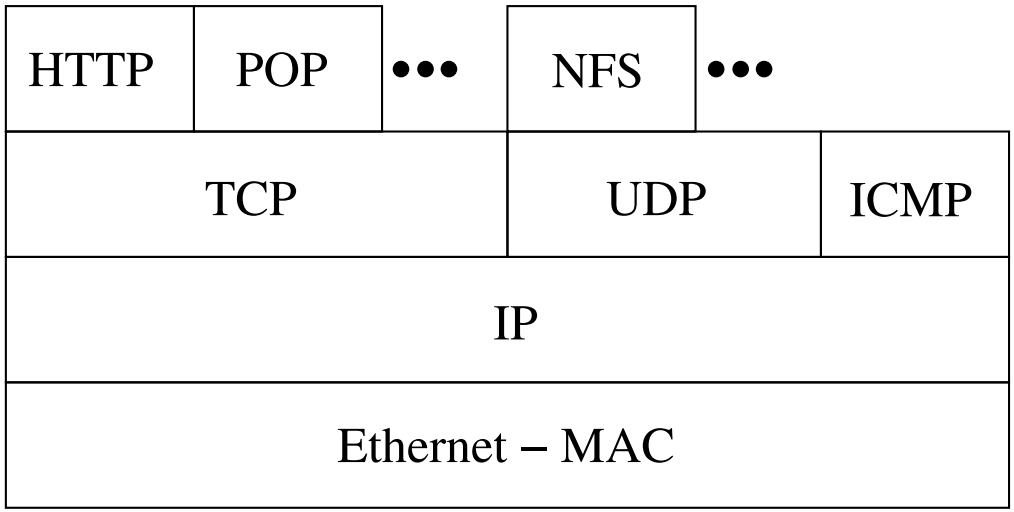
- `whois <objet>` parmi
  - domaine DNS
  - serveur de noms (NS)
  - système autonome (ex. AS12322)
  - adresse IP $\longrightarrow$ AS
  - ... (18 types d'objet)

- Références
- RFC 954, RFC 3912 (cf Bortzmeyer)

Configuration réseau "intelligente" (intranet)

- **À éviter pour les serveurs**
- *Avahi*
  - Implémentation libre de Zeroconf (Apple *Bonjour*)
  - adresses IPv4 Link-local 169.254.0.0/16
  - DNS-SD : découverte automatique de services (impression...)
  - mDNS (multicast) : 224.0.0251 ou ff02 : :00fb
  - service `avahi-daemon` + bibliothèque `libavahi`
- *Network Manager*
  - surcouche de configuration "intelligente" du réseau
  - active la meilleure connexion disponible (câble, wifi...)
  - service `network-manager`
  - interface graphique (KDE) ou applet (Gnome) ou CLI

Retour sur la pile IP



Passage aux couches supérieures des protocoles (hors ICMP)

Services et ports *109.3*

Service

Programme côté serveur dans une relation client/serveur

Attaché à un couple port/protocole

La référence : /etc/services

Liste **informative** des services communs

Ports

- désigné par un numéro entre 0 et 65535
- attaché à un protocole : 43/TCP $\neq$ 43/UDP
- les ports 1 à 1023 sont réservés à root
- normalisés par l'IANA
  `http://www.iana.org/assignments/port-numbers`

netstat : diagnostic des connexions et services *109.3*

- Modes de fonctionnement
  - interfaces `--interfaces | -i`
  - routes `-route | -r`
  - **connexions établies** (ip ou sockets unix) `--ip | --unix`
  - **services à l'écoute** `--ip -l`
  - statistiques (`-s`)
  - groupes multicast (`-g`)
  - masquerading (`-M`)

- Options globales (ou presque)

  `-p` programme + PID (root seulement)
  `-c` en continu (toutes les secondes)
  `-n` numérique (port ou adresse)
  `-e` (extra) compléments (User, Inode)

Focus : les états TCP

109.3

- Établissement de connexion

  LISTEN état normal d'attente
  SYN-SENT
  SYN-RECEIVED

- Connexion établie

  ESTABLISHED état normal de connexion

- Fin de connexion

  FIN-WAIT-1
  FIN-WAIT-2
  CLOSE-WAIT
  LAST-ACK
  TIME-WAIT maxi. 4 minutes
  CLOSED

inetd : le super-démon *110.2*

Mode d'exécution d'un service

- démon : lancé indépendamment (`/etc/init.d/`)
- `inetd` : lancé à la demande par le super-démon openbsd-inetd

Exemple : telnet

- Installer `telnet` et `telnetd`
- `netstat -a -tu -ep` avec et sans connexion `telnet`
- configuration dans `/etc/inetd.conf`
- Désinstaller telnetd !

Compléments et variantes

`xinetd` remplace fréquemment `inetd`.

tcpwrapper *110.2*

Deux modes de fonctionnement

- démon `tcpd`, invoqué par `inetd`
- bibliothèque `libwrap` liée à certains serveurs (ex. sshd)

Son rôle : sécurisation

- Contrôle des autorisations
- Configuration :
  - `/etc/hosts.allow`
  - `/etc/hosts.deny` in.telnetd :ALL

Pour aller plus loin

- `tcpdmatch` et `tcpdchk` : tests et débogage des règles
- `man hosts_access` et `man tcpd`

xinetd : l'alternative *110.2*

- Principes
  - plus générique et plus complet : un fichier par service
  - par défaut sous RedHat
- Configuration
  - `/etc/xinetd.conf` : configuration globale
  - `/etc/xinetd.d` : un fichier par service (cf `/etc/services`)
- Principales règles

instances	nombre maximal d'instances simultanées
log_type	syslog, fichier, etc.
cps	nombre maximal de connexions par seconde
user	propriétaire du processus
only_from	restriction d'accès
access_times	restrictions temporelles

Sécurité et diagnostic

Diagnostic des protocoles texte clair

- Les commandes disponibles
  - `telnet` client texte bas-niveau
  - `telnetd` serveur protocole TELNET
  - `netcat` (`nc`) alternative plus bas niveau

- Session `telnet <hote> <port>`

```
$ telnet cressida 80
Connected to cressida.localnet.
Escape character is '^]'.
GET /
<html><body><h1>It works!</h1></body></html>
Connection closed by foreign host.
```

Diagnostic des protocoles texte sur SSL/TLS

- `openssl` : utilitaire générique SSL/TLS
  - création de paramètres des clefs RSA, DH et DSA
  - création de certificats X.509, CSRs et CRLs
  - calcul de condensés de messages
  - chiffrement et le déchiffrement
  - test de clients et serveurs SSL/TLS
  - gestion de courriers S/MIME signés ou chiffrés

- Session `openssl s_client`

```
$ openssl s_client -connect cressida:443
CONNECTED(00000003)
depth=0 /CN=cressida.localnet
[...]
GET /
<html><body><h1>It works!</h1></body></html>
closed
```

Performances réseau et bande passante

- Surveillance instantanée
  - Commande `iftop` : capture au vol
  - Utilitaire `iptraf` : interface semi-graphique
  - Utilitaire `slurm`
  - Utilitaire `bmon`
- Supervision long terme : serveur `ntop`
  - sonde et collecte
  - interface web

tcpdump & wireshark

Outils pour examiner les données en transit

- `tcpdump` Interception simple en mode texte
- `wireshark` Interception avancée en mode graphique
  Filtrage à l'acquisition (libpcap)
  Filtrage à l'affichage
- `tshark` : équivalents en mode texte

Exemples

Requêtes DHCP, DNS, connexion web, etc...

Des dangers de la promiscuité...

Une carte ethernet peut passer en mode *promiscuous*
$\longrightarrow$ elle examine alors tous les paquets de son réseau physique
Exemple : `tcpdump dst net 192.168.0.123` espionne cette IP
attention équipement : hub, switch, switch "manageable"

tcpdump & wireshark - filtres

- Filtres à l'acquisition (libpcap)

- Filtres à l'affichage

nmap : un scanner de ports *110.1*

Utilisation

- local : idem `netstat` + `unhide-tcp`
- diagnostic
  `nmap -sP <network>` : émule un ping Broadcast
- attaque réseau
  `nmap -sT <host>` : trouver les ports TCP ouverts sur *host*
- attaque réseau
  `nmap -sS <host>` : idem, mais plus discret

Remarques

- Certaines options (-sS) nécessitent d'être root
- Attention, pas de geste déplacé !

Pare-feu : Netfilter + IPtables

- Deux types de pare-feux
  - monoposte (à la Windows)
  - équipement réseau dédié (plusieurs interfaces réseau)

- Architecture
  - `netfilter` : en espace noyau
  - des modules `ipt_*` : extensions
  - commandes `iptables` et `ip6tables`
  - `arptables` : filtrage ARP (ethernet)
  - `ebtables` : *ethernet bridging*

- Des interfaces utilisateurs “conviviales”
  - `firestarter` : interface graphique “monoposte”
  - `fwbuilder` : interface graphique “serveur” (plusieurs backends)
  - `shorewall` : sur-couche d'abstraction (classes de machines...)
  - ...

IPtables : introduction aux concepts

- Trois tables
  - `filter` : règles de filtrage (accepter, refuser... un paquet)
  - `nat` : modification des IP et ports source ou destination
  - `mangle` : modification des paramètres et contenu des paquets

- Cinq chaînes correspondant aux “embranchements”
  - INPUT : concerne les paquets destinés au pare-feu
  - OUTPUT : concerne les paquets émis par le pare-feu
  - FORWARD : concerne les paquets transitant par le pare-feu
  - PREROUTING : s'applique aux paquets dès qu'ils arrivent
  - POSTROUTING : s'applique aux paquets prêts à partir
  - ... (définies par l'administrateur)

- Des actions (en fonction des tables et des chaînes) : REJECT, DROP, ACCEPT, LOG...

IPtables - concepts 2

- Relations tables - chaînes

	filter	nat	mangle
INPUT	X		X
OUTPUT	X		X
FORWARD	X	X	X
PREROUTING		X	X
POSTROUTING		X	X

- Embranchements

IPtables - exemples

Exemple de filtrage (eth0=LAN eth1=Internet)

```
iptables -t filter -P FORWARD DROP
iptables -t filter -A FORWARD -i eth0 -p tcp --dport 80 -j ACCEP
iptables -t filter -A FORWARD -i eth1 -p tcp --sport 80 -j ACCEP
```

Exemple de NAT (traduction d'adresse)

```
iptables -t nat -A PREROUTING -i eth1 -p tcp --dport 80 \
        -j DNAT --to-destination 192.168.1.3:8080
```

Protection contre les attaques SSH

(pas plus de 2 tentatives SSH par minute et par IP)

```
iptables -I INPUT -p tcp --dport 22 -i eth0 -m state --state NEW
 -m recent --set
iptables -I INPUT -p tcp --dport 22 -i eth0 -m state --state NEW
-m recent --update --seconds 60 --hitcount 3 -j DROP
```

Autres usages d'IPtables

- Comptabilité IP (IP Accounting)
  - Mesure de la bande passante utilisée
  - par adresse (source ou destination)
  - par port (=service)
  - par protocole (ICMP, TCP, UDP)...
-

iptables : filter

Exemple simple

- Couper tout envoi
  `iptables -t filter -P OUTPUT DROP  couper tout envoi`
- Autoriser les envois vers soi-même
  `iptables -t filter -A OUTPUT -s 127.0.0.0/8 -o lo -j ACCEPT`
- Autoriser les envois HTTP `iptables -t filter -A OUTPUT -s 127.0.0.0/8 -p tcp -dport 80 -j ACCEPT`

Travaux pratiques

- Tester la connexion au SMTP local avec telnet
- Comparer service activé, service désactivé
- Mettre en place une règle de pare-feu ACCEPT
- Comparer les effets des cibles DROP et REJECT

Wifi

Presque comme l'ethernet câblé !

Carte wifi (802.11b/g) : carte ethernet particulière

- adresse MAC similaire (6 octets)
- nom d'interface variable (eth1, wlan0, wifi0, ath0...)
- nécessite un point d'accès (serveur DHCP)

Configuration manuelle

- ifconfig wifi0 up
- iwlist wifi0 scan
- iwconfig wifi0 essid *[key <clé>]*
- dhclient wifi0

Sécurité

Risques supplémentaires par rapport au cablé

Surcouches de sécurité : VPN, portail captif, etc.

Sauvegarde et archivage

Sauvegarde et archivage

Rappel : archives

tar (archivage) + gzip / bzip2 (compression)

Sauvegardes

- Historiquement, sur bandes $\Longrightarrow$ accès longs, séquentiels
- images (disque ou partition) : `dd`, `partimage`, `Clonezilla`
- `dump + restore` : outil Unix historique de sauvegarde, orienté bandes
- `cpio` : alternative à tar
- `rsync` : commande orientée synchronisation (locale ou distante)

Applications complètes

- Amanda : disques + bandes, ligne de commande
- BackupPC : disques seulement, interface web
- ...

TP - Sauvegarde et archivage

dump + restore

- sauvegarde totale de `/etc` avec `dump`
- restauration interactive de `fstab` et `modprobe.d` dans `/mnt/vol/etc`

- sauvegarde d'un système de fichier au niveau 0 (complète)
- modification de quelques fichiers
- sauvegarde incrémentale des différences
- restauration complète

rsync : synchronisation de répertoires

- Modes de transfert
  - push : le client envoie ses données
  - pull : le serveur récupère les données ciblées
- Protocoles réseau utilisables
  - local
  - ssh
  - rsh
  - rsyncd : démon et protocole spécifique
- Fondation : librsync
  - calcul efficace des différences entre binaires
  - algorithme “rolling checksum”

Compléments à rsync / librsync

- Idée : “snapshots” (images...)
  - sauvegardes incrémentales via rsync
  - liens durs pour compléter
- Solutions légères
  - rdiff-backup (python) : push+pull
  - rsnapshot (perl) : pull
  - dirvish (perl) : pull
  - rlbackup (C) : push (vise la sécurité)
- Applications
  - BackupPC (perl) : interface web

Compléments : suivi de version et réplication

Suivi de version

Pour les fichiers sensibles, par exemple `/etc/`

- Principe : stocker l'historique des versions successives
- Outils : CVS, Subversion, SVK...

Réplication

Pour la sécurité et l'intégrité des données, la redondance

- les fichiers de log (via rsyslog, syslog-ng...)
- les bases de données (serveurs maître et esclaves)
- les annuaires (LDAP...)

Impression réseau sous Unix

L'impression sous Unix

Matériel : 3 types de connexions

- imprimantes locales (// ou USB)
- imprimantes réseau (interface ethernet)
- imprimantes locales sur un serveur d'impression (réseau)

Services et protocoles

- applicatif : prépondérance de **PostScript** puis **PDF** (Adobe)
- **lpd/lpr** : historique, RFC1179, 1990
  - **lpd BSD** : implémentation historique
  - **LPRng** : réécriture du précédent (RH)
- **CUPS** : Common Unix Printing System
  - RFC 2565-2569, 1999 (Novell - Xerox)
  - Easy Software Products (1997-2007), puis Apple
  - protocole IPP, surcouche à HTTP
  - configuration service inspirée d'Apache

Configuration de l'impression

LPD / LPRng

- un démon : `lpd` (TCP port 515)
- un fichier de configuration : `/etc/printcap`
- des commandes : (BSD) `lpr, lpq, lprm, lpc` ou (SystemV) `lp, lpstat, cancel, lpadmin`

CUPS

- un démon : `cupsd` (TCP ports 515 et 631)
- interface web : `http://localhost:631`
- un répertoire de configuration : `/etc/cups/*`
- paquets Debian : `cupsys, cupsys-bsd...`
- surcouches graphiques :
  - GNOME : gnome-cups-manager
  - KDE : kdeprint (uniformise l'accès aux 3 systèmes)

En pratique : CUPS

- Installation (paquets)
  - (deb) `cups`, `cups-common`, `cups-client`, `cups-bsd`
  - (RH) `cups`
- Fichiers
  - Configuration `/etc/cups/...`
    `cupsd.conf` configuration du service
    `printers.conf` configuration des imprimantes
    `ppd/*` Postscript Printer Description
  - Travaux `/var/spool/cups`, `/var/cache/cups/*`
  - Logs `/var/log/cups` (cupsd)
- Références
  - Linux Foundation - *OpenPrinting*
  - Wikipedia, article CUPS

Le chiffrement personnel avec GPG

De PGP à GPG

110.3

- Historique
  - 1991 Phil Zimmermann crée PGP (Pretty Good Privacy)
  - 1997 OpenPGP (standard) devient la RFC 2440 (IETF)
  - 1999 Gnu Privacy Guard (GnuPG / GPG)
  - 2003- *G10code* fondée par Werner Koch, principal développeur

- Technologies utilisées par GPG
  - compression de fichiers
  - hachage (cryptographique)
  - cryptographie symétrique
  - cryptographie à clés publiques

GPG : interfaces utilisateurs *110.3*

- Ligne de commande
  - gpg couteau-suisse complet, chiffrement et signature
  - gpgv uniquement vérification de signature
  - paquets nécessaires : gnupg, (gnupg-agent, pinentry)
- Interfaces graphiques générales
  - *Seahorse*, frontal Gnome
  - *KGPG*, frontal KDE
  - *GnuPG Shell*, interface multi-plateformes
- Intégrations multiples
  - mail : Kmail (KDE), Evolution (Gnome), plugin Enigmail (Thunderbird...)
  - messagerie instantanée (Gajim...)
  - éditeurs de texte : vim, emacs...

En pratique : chiffrer un fichier *110.3*

- Créer une clé personnelle
  - `gpg --gen-key`, puis
  - `gpg (--armor) --output maclef.pub.asc --export`
- Chiffrer un fichier
  - clé publique : `gpg --recipient qui@mail.domain --encrypt msg.txt --output msg.gpg`
  - symétrique : `gpg --symmetric msg.txt --output msg-sym.gpg`
- Déchiffrer un fichier
  - clé privée : `gpg --decrypt msg.gpg --output message.txt`
  - symétrique : `gpg --decrypt msg-sym.gpg --output message.txt`

En pratique : signer et authentifier un fichier *110.3*

- Signer un fichier : 3 modalités
  - `gpg --sign fichier` ⟶ `fichier.gpg` (binaire)
  - `gpg --clearsign fichier` ⟶ `fichier.asc` (texte)
  - `gpg --detach-sign fichier` ⟶ + `fichier.sig` (annexe)
- Extraire et vérifier un fichier signé
  - Vérifier : `gpg --verify fichier.[gpg|asc|sig]`
  - Extraire : `gpg --decrypt fichier.[gpg|asc]`

GPG et *web of trust* *110.3*

- Importer une clé
  - fichier : `gpg --import pubkey.asc`
  - `gpg --listkeys | --list-secret-keys`
  - depuis un serveur : `gpg --search-keys <qui@mail.domain>` ou `gpg --recv-keys`
- Signer une clé
  - `gpg --sign-key <clef-id>`
  - cf `gpg --list-sigs` et `gpg --check-sigs`
- Envoyer la clé sur un serveur
  - `gpg --send-key <clef-id>`

Autour de PGP / GPG

110.3

- Solutions d'authentification
  - OpenPGP Card (carte à puce)
  - poldi (pam + openpgp smartcard)
  - ...
- Bibliothèques
  - libgcrypt : bibliothèque principale de GPG
  - autres : bibliothèqu Gnu TLS