Formation Unix/Linux - administration

Guillaume Allègre
Guillaume.Allegre@silecs.info

Grenoble INP - Formation Continue

2015

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/

0

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 4.0 (février 2015) : `http://wiki.lpi.org/wiki/LPIC-1_Objectives_V4(FR)`

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.

Partie 1/2 - Administration système

- Rappels
- Gestion des paquets (Debian/Redhat)
- Gestion des fichiers de configuration
- Gestion des utilisateurs
- Gestion des services
- Gestion des logs
- Administration des ressources

Partie 2/2 - Systèmes de fichiers et réseaux

- Systèmes de fichiers
- Réseau
  - Configuration réseau
  - SSH et utilitaires
  - NTP : base de temps réseau
- Services spécifiques
  - X-Window (X11)
  - CUPS
  - Sauvegarde
  - Archivage

Gestion des paquets (deb et rpm)

Gestionnaires de paquets

- Toutes les distributions (ou presque)
  - Paquets sources / **binaires**
  - Deux niveaux de gestion des paquets
    - bas niveau : paquet individuel
    - haut niveau : dépôts, dépendances

- En pratique

	bas niveau	haut niveau	++
Debian	dpkg	(APT)	aptitude
Ubuntu	dpkg	synaptic (APT)	(aptitude)
Redhat	rpm	yum	-
Mandriva	rpm	urpmi	-
SuSE	rpm	YAST	-

Gestion des paquets avec APT

La famille apt

102.4

- synaptic
- aptitude
- apt-get
  - update
  - install
  - ...
- apt-cache
  - search
  - show
  - policy

Fichiers

`/etc/apt/apt.conf.d/`

`/etc/apt/sources.list`

Documentation

`apt-howto-en`, `apt-howto-fr`

TP – apt

102.4

1. Mettre à jour sa distribution.
2. Examiner le fichier `/etc/apt/sources.list` et en comprendre la syntaxe.
   Quelle est l'organisation d'un miroir Debian ?
3. Ajouter aux sources APT les dépôts de la distribution testing.
   Que se passe-t-il en cas de demande de mise à jour ?
4. Créer `/etc/apt/apt.conf` afin de fixer la version (*release*) par défaut à stable.
   Retenter une mise à jour.
5. **apt** garde une copie de sauvegarde des paquets téléchargés. Effacer ces fichiers.

La dernière évolution : aptitude *102.4*

- Historique
  1. dselect
  2. apt-get
  3. aptitude

- Interfaces
  - Ligne de commande (sous-commandes compatibles `apt-get`)
  - Interface semi-graphique (ncurses)

- Les avancées d'aptitude
  - un log des opérations : `/var/log/aptitude`
  - distinction paquets : installés automatiquement / à la demande
  - résolution des dépendances : meilleure, plusieurs alternatives

- Documentation : `aptitude-doc-en`, `aptitude-doc-fr`

Les paquets Debian *102.4*

Paquet binaire (.deb) ou source (.dsc)

Contenu d'un paquet binaire (.deb)

- Archive des fichiers (data)
- Métadonnées (control/control)
  - Descriptions textuelles : courte et longue
  - Section : classement du paquet dans une hiérarchie debian
  - Version
  - Dépendances, conflits, suggestions, recommandations...
  - debtags : indexation du paquet
    Par exemple : network::service, suite::apache
- Utilitaires (control/...)
  - scripts installation / suppression
  - sommes de contrôle (MD5sum)

Travaux pratiques

102.4

Examen du paquet dpkg

- À la main
  Commandes : `ar t`, `tar -x`
- Avec l'outil dédié
  Commande : `dpkg-deb`

Deux cas particuliers *102.4*

- Méta-paquets
  - Paquet “réel” : le .deb existe
  - Paquet de paquets : via les dépendances
  - Exemple : `gnome`
- Paquets virtuels
  - Paquet virtuel : le .deb n'existe pas
  - Indique un service générique, fourni par plusieurs paquets
  - Exemple : `mail-transport-agent` ; cf `mailman`

dpkg : gestion locale

102.4

dpkg manipule les paquets debian (.deb) sans accès réseau.

Principales options de dpkg

- `dpkg -i paquet.deb` ⟶ installe
- `dpkg -r paquet` ⟶ désinstalle
- `dpkg -L paquet` ⟶ liste les fichiers du paquet
- `dpkg -S fichier` ⟶ recherche `fichier` parmi les paquets installés

dpkg est souvent nécessaire pour les opérations fines (conflits importants, diagnostic, etc.)

TP – dpkg

102.4

1. Installer `ncdu` à partir des sources. En quoi est-ce pénible ?
2. Télécharger le navigateur Opera (`www.opera.com`) et l'installer grâce à `dpkg`.
3. Avec `dpkg`, lister les fichiers installés par Opera.
4. Quels exécutables sont fournis par le paquet **sysvinit** ?
5. Quels sont les paquets actuellement installés sur votre machine ?
6. De quel paquet provient la commande `ifconfig` ?
7. Reconfigurer le serveur mail local.

Reconfiguration d'un paquet *102.4*

Debconf

- une mémoire des choix de configuration
- interfaces : dialog, readline, n-i, gnome, kde, (editor, web)
- priorités : low, medium, high, critical
- fichier de configuration : `/etc/apt/apt.conf.d/70debconf`
- base : `/etc/debconf.conf`, `/var/cache/debconf/*`
- manpages : `debconf(7)`, `debconf(1)`, `debconf.conf(5)`

Commandes

- `dpkg-reconfigure <paquet>`
- manpages : `dpkg-reconfigure(8)`, `dpkg-preconfigure(8)`

Le suivi de bugs de Debian

102.4

BTS : le Bug Tracking System

- `http://www.debian.org/Bugs/`
- intégration à APT : `apt-listbugs`

Déposer un bug

- le paquet `reportbug`

Gestion des paquets RPM avec Yum

Les paquets RPM

102.5

- Paquet binaire (RPM) ou source (SRPM)
- Archive de fichiers : `rpm -ql pam`
- Description textuelle courte et longue
- Métadonnées `rpm -qi pam`
  - Version
  - Date et auteur de la compilation
  - Groupe (classification)
  - Références upstream
  - Informations dépendances

Commande rpm : gestionnaire global *102.5*

`rpm` manipule les paquets redhat (`.rpm`) sans accès réseau.

Information sur un paquet installé

- `rpm -qi paquet` ⟶ métadonnées
- `rpm -ql [ |-c|-d] paquet` ⟶ contenu de l'archive
- `rpm -q -whatprovides paquet` ⟶ dépendances...
- `rpm -qf fichier` ⟶ recherche `fichier` parmi les paquets installés

- À quoi sert le paquet `pam` ?
- Quels sont les paquets actuellement installés sur votre machine ?
- De quel paquet provient la commande `ifconfig` ?

Commande rpm : installer, supprimer, vérifier des paquets

102.5

Installation, mise à jour, suppresion

- `rpm -ivh paquet.rpm` : installe un fichier-paquet
- `rpm -[U|F]vh paquet.rpm` : met à jour un fichier-paquet
- `rpm -[e] paquet` : supprime un paquet

Vérification d'un paquet

- `rpm -qV paquet` $\longrightarrow$ compare à l'état initial

RHN - RedHat Network

102.5

- `up2date`
  - `-u` : update
  - `-i` : install
  - `-show-available`
  - `-register` : RHN canonique

- `yum`
  - plus puissant
  - disponible mais non standard en RHEL 4.x
  - remplace `up2date` en RHEL 5.x

Fichiers

`/etc/sysconfig/rhn/up2date`
`/etc/sysconfig/rhn/sources`

TP – utiliser le réseau RHN

102.5

1. Mettre à jour sa distribution.
2. Examiner le fichier `/etc/sysconfig/rhn/sources` et en comprendre la syntaxe.
   Quelle est l'organisation d'un dépôt CentOS ?
3. Examiner le fichier `/var/log/up2date`
4. Installer `xpdf` avec `yum` ou `up2date`
5. Trouver une source externe pour installer `iftop` :
   `http://dag.wieers.com/rpm`

Autres systèmes d'installation

Attention aux conflits !

- Système fourni par une application
  - mise à jour automatique : ex. Firefox
  - extensions et plug-ins : ex. Firefox !
- Gestionnaire de paquet "upstream"
  - CTAN : Comprehensive TEX Archive Network (1992)
  - CPAN : Comprehensive Perl Archive Network
  - PEAR / PECL : extensions PHP
  - PyPI : Python Package Index
  - RubyGems ...

Pour aller plus loin...

- Modifier des paquets existants (mises à jour...)
  - Récupérer le paquet source et le modifier
  - Recompiler les paquets binaires
- Créer ses propres paquets
- Maintenir un dépôt miroir local (ou un proxy)
- Maintenir un dépôt local (paquets locaux)

Gestion de la configuration

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

Pour aller plus loin

Gestionnaire de configuration

- Gestion de grands parcs de serveurs
- Définition centralisée, basée sur des règles
- Déploiement automatisé

Les principaux compétiteurs

- CFEngine
- Chef
- Puppet

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 - 1/2 *110.1*

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

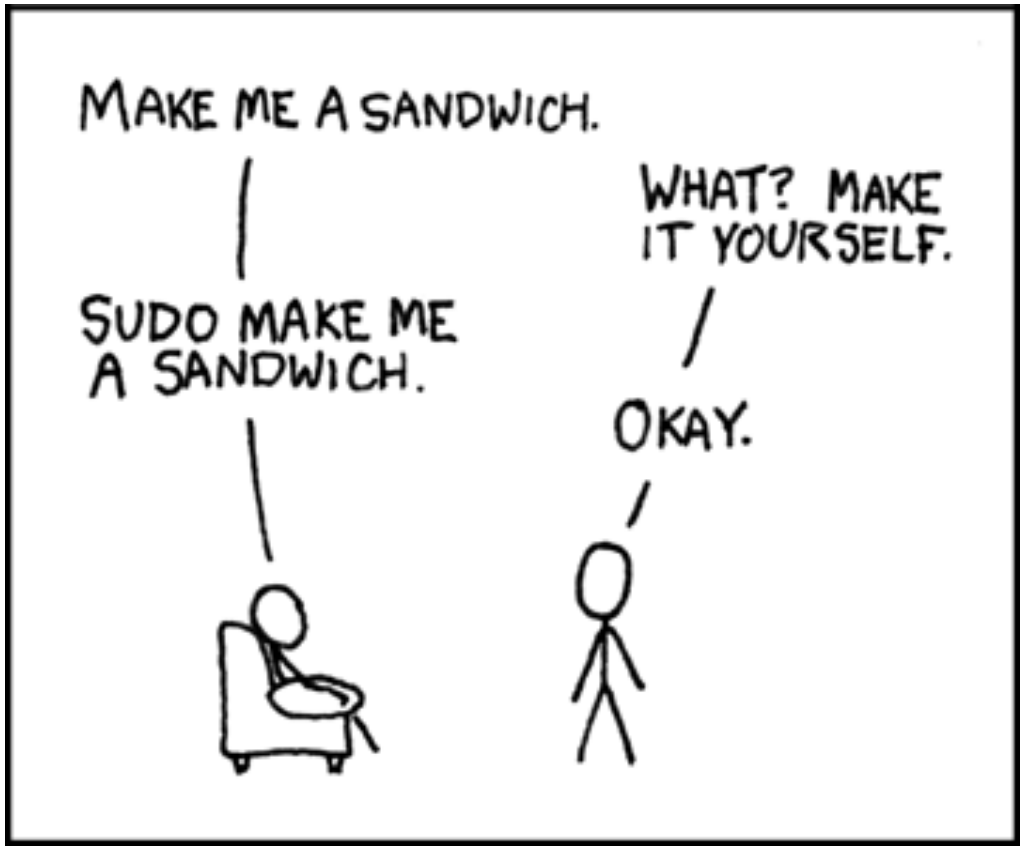
- En pratique
  - auxiliaire : `visudo [-c]` modifie et vérifie le fichier `sudoers`
  - documentation : `man sudoers`

Les sudoers - 2/2

110.1

- Les commandes utilisateurs :
  - `sudo (-u <u-cible>) <commande>`
  - `sudoedit <fichier>` ou `sudo -e <fichier>`
- Les traces
  - voir `/var/log/auth.log` (syslog)

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.

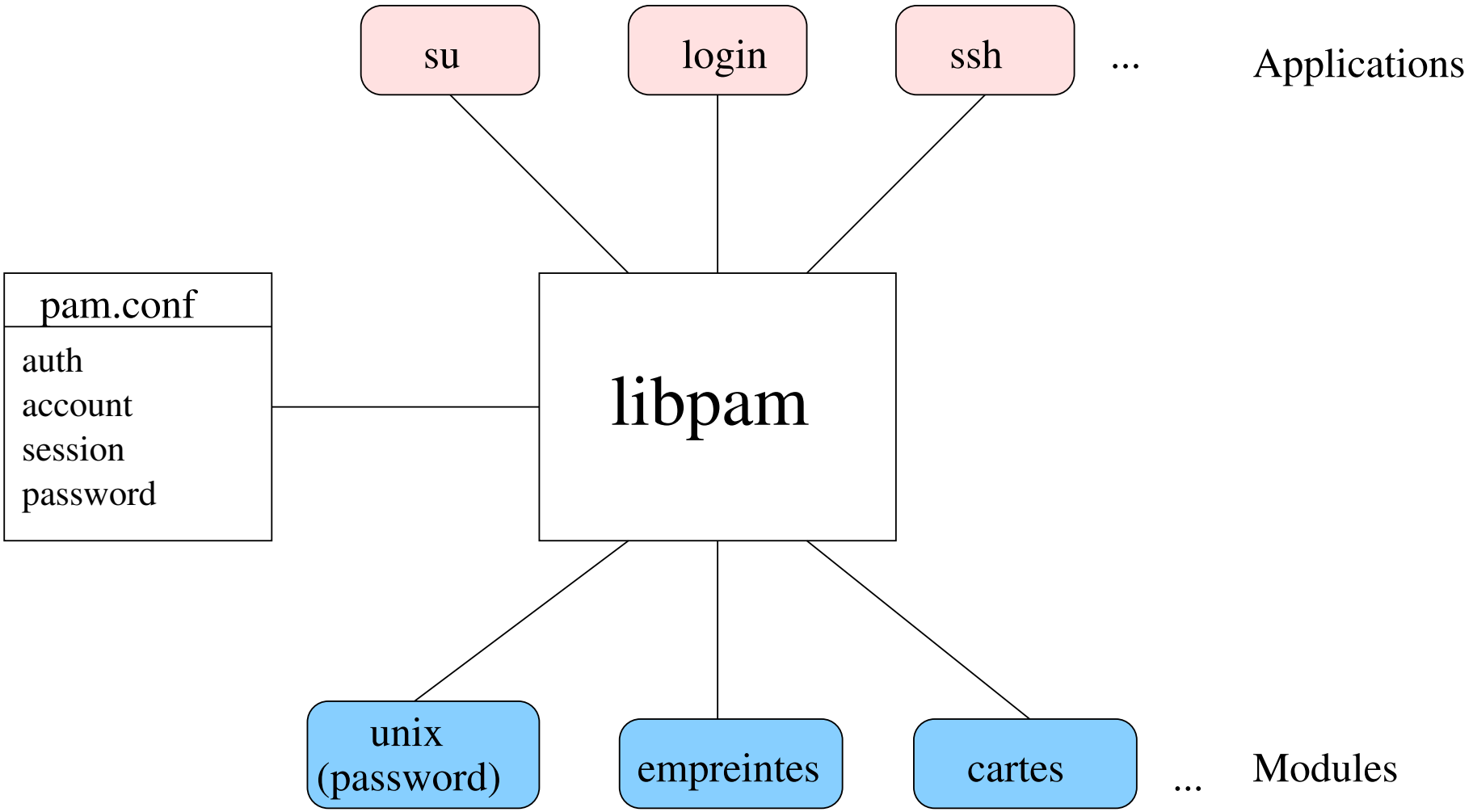
PAM : Pluggable Authentication Modules

- 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

Boot système

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. Transition vers le mode utilisateur
   - SystemV init basé sur inittab et les runlevels
   - ou systemd (récent) propre à Linux
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 - Menu

102.2

- Esc menu standard
- C CLI mode : micro-shell style Bash
  ex. `grub> cat (hd0,3)/etc/fstab`
- E Edit mode : entrée de menu courante

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 ?

Systemd

Systemd - Fonctionnalités principales *101.3*

Remplacement complet de SysVinit

- compatible (englobe) sysVinit et les scripts LSB init
- forte parallélisation et dépendances entre services
- spécifique à Linux (cgroups, notify)
- limite fortement les scripts shell
- `Rethinking PID 1`, Lennart Poettering, 30 avril 2010

Innovations

- intègre surveillance / relance des processus
- intègre la supervision distante (via ssh)
- accès unifiés aux logs service
- architecture client / serveur : démon `systemd`

Mise en place *101.3*

Sous Debian

- par défaut à partir de Jessie (8.0, avril 2015)
- paquets `systemd`, `systemd-sysv` (compatibilité)
- optionnel : `systemd-ui` interface graphique `systemadm`
- mise en oeuvre partielle, sysVinit reste possible

Sous RedHat

- par défaut à partir de RHEL 7.0 (juin 2014)
- bascule totale vers systemd

Les unités systemd *101.3*

L'élément de base de la configuration de systemd

Les unités : + 12 catégories (suffixes)

- service `.service` ex. `sshd.service`
- cible `.target` groupe de services (cf infra)
- point de montage `.mount` ex. `home.mount`
- socket `.socket` ex. `sshd.socket`
- ...

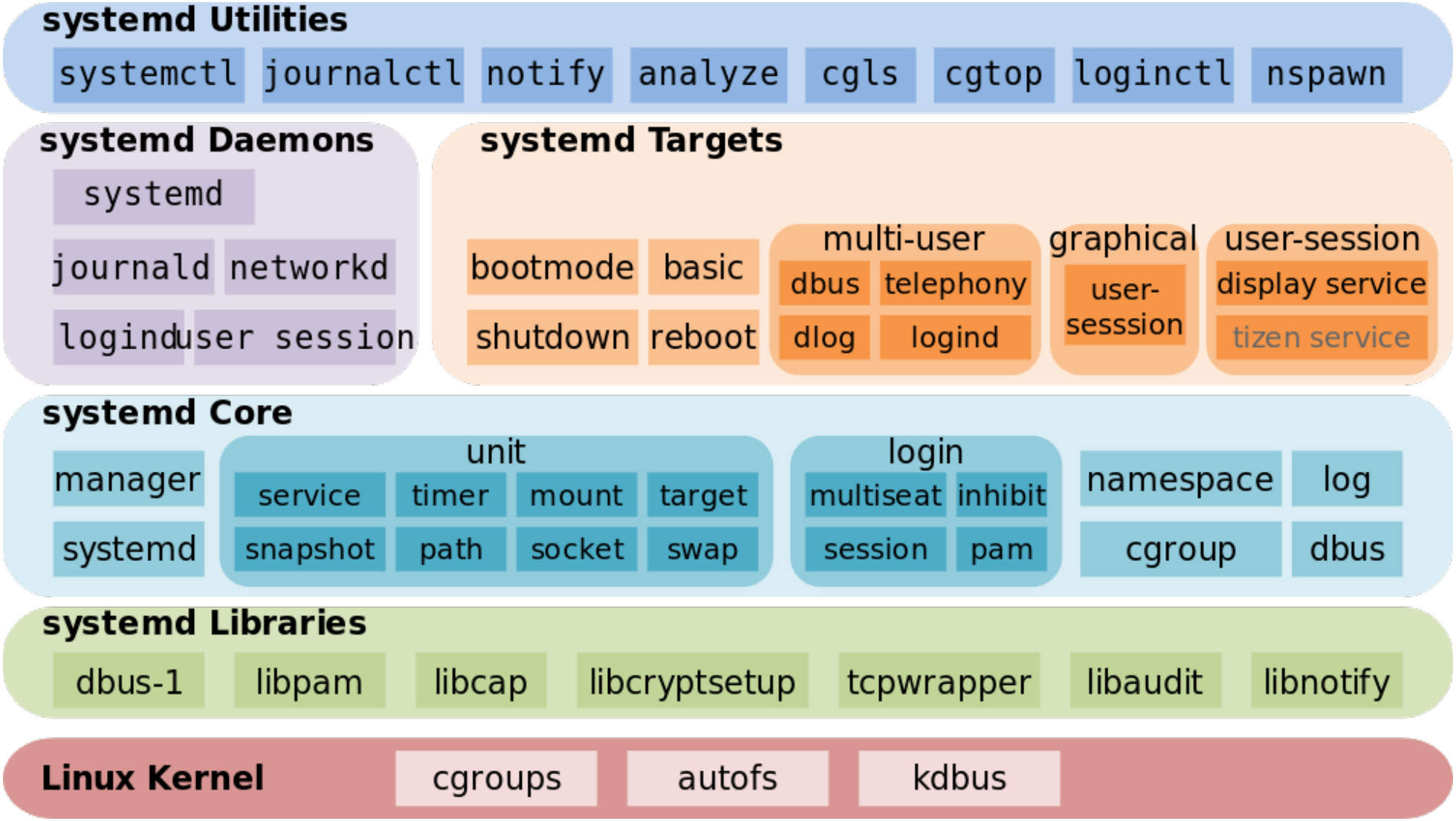
Emplacements des fichiers

- `/lib/systemd/system` installés par les paquets
- `/run/systemd/system` installés par l'autodétection
- `/etc/systemd/system` installés par l'administrateur

Exemples

- fichiers `sshd.socket` et `sshd.service`
- syntaxe .ini
- `man systemd.unit`

Panorama général de Systemd



Shmuel Csaba Otto Traian, CC BY-SA 3.0 ou GFDL via Wikimedia Commons - Systemd_components.svg

Commandes de diagnostic *101.3*

Liste des unités

- `systemctl list-units [--type service] [--all]`
- `systemctl list-unit-files`

Détails

- `systemctl status <unité>`
- `systemctl is-active <unité>`
- `systemctl show <unité>` toutes les propriétés
- `systemctl is-enabled <unité>`

Aide et documentation

- `systemctl help <unité>` documentation unifiée
- `systemctl -help` aide sur systemctl

Commandes d'action immédiate *101.3*

Lancement des services

- `systemctl start <unité>`
- `systemctl stop <unité>`
- `systemctl restart <unité>`
- `systemctl try-restart <unité>`
- `systemctl reload <unité>`
- par exemple avec `cron.service` ou `bluetooth.service`

À comparer

- `service <service> start` (unifié sysVinit)
- `/etc/init.d/<script> start` (historique sysVinit)

Commandes de configuration *101.3*

Activer le démarrage au boot

- `systemctl enable <unité>`
- `systemctl disable <unité>`
- `systemctl reenable <unité>`
- crée / supprime les liens symboliques dans `/etc/systemd/system/`

Masquer des services

- `systemctl mask <unité>`
- `systemctl unmask <unité>`
- crée / supprime un lien symbolique vers `/dev/null`
- empecher tout démarrage manuel ou par dépendance

Supervision du démon systemd

- `systemctl daemon-reload` recherche les unités nouvelles ou modifiées
-

Cibles systemd

101.3

Contexte

- les cibles (targets) remplacent les runlevels de sysVinit
- unité cible = groupe d'autres unités (cibles, services...)
- ex. `graphical.target` regroupe `multi-user.target`, `display-manager.target`

Configuration

- `systemctl get-default`
- `systemctl list-units - -type=target`
- `systemctl set-default multi-user.target`

Cibles et mode de dépannage *101.3*

Opérations spéciales

- `systemctl isolate <cible.target>`
- équivalent à un changement de runlevel sysVinit

Modes de dépannage

- `systemctl rescue` — single-user mode
- `systemctl emergency` — single-user minimaliste

Alimentation

101.3

Modes classiques (serveur)

- `systemctl halt`
- `systemctl poweroff`
- `systemctl reboot`

Modes mobiles (économie d'énergie)

- `systemctl suspend` — en RAM
- `systemctl hibernate` — sur disque
- `systemctl hybrid-sleep` — les deux
- remplace les commandes `pm-*`

Planification des tâches

Les services : cron

`crond` : lancement périodique de tâches

- `crond` démon (résident) qui réalise les tâches de fond du système.
- granularité = 1 minute
- email sortie utilisateur

`crontab` : les tables de tâches

- Les crontab utilisateurs (dont root)
- 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 !

Gestion des logs

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

Client CLI : `logger`

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

toutes facilities *sauf kernel*

tester avec `auth` + `emergency` puis `auth` + `debug`

Fichier de configuration syslog *108.2*

- 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`

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`

Systemd journal - 1

108.2

- Configuration
  - `/etc/systemd/journal.conf`
  - ex. `Storage = auto | persistent | volatile`
- Stockage des logs
  - `/run/systemd/journal/*` volatil
  - `/var/log/journal/*` pérenne
  - stockage binaire (métadonnées) + texte
- Exercice
  - trouver le démon "journald"
  - trouver ses fichiers, sockets...

Systemd journal - 2

108.2

- Consultation
  - commande `journalctl`
  - utilisateur `root` pour les journaux système
- Paramètres
  - reboots : `journalctl -b 0, -b -1 ...`
  - horodatage : `journalctl --since="2015-05-30 12:34:56" --until...`
  - formatage : `journalctl -o short, short-iso, verbose, json...`
  - unité : `journalctl --unit=ssh`
  - processus : `journalctl _PID=12345`

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 ressources

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`

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 - 1/2

- `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

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 processus et exécutables - 2/2

- `ltrace` : tracer les appels de bibliothèques
  - `ltrace -l <bibli>` → limite la trace à cette bibliothèque
  - configuration : `/etc/ltrace.conf`

Exo

1. 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

Administration des Systèmes de fichiers

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

Système de fichiers par défaut : ext2 / ext3 / ext4

- `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 "disque" *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=cdrom.iso`
2. monter localement l'image dans `/media/image`
   `mount -t iso9660 [-o loop] cdrom.iso /media/image`
3. vérifier les boucles avec `losetup -a`
4. (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)

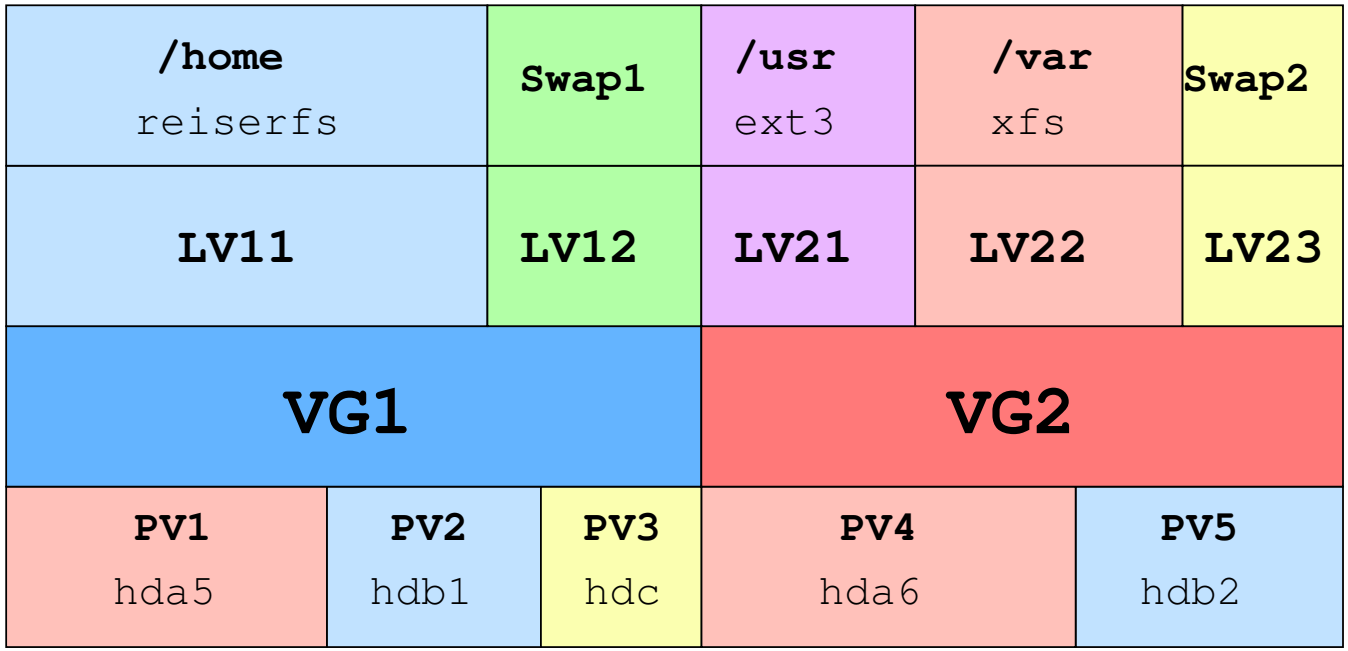
Périphériques blocs virtuels
LVM, raid...

Périphériques blocs virtuels

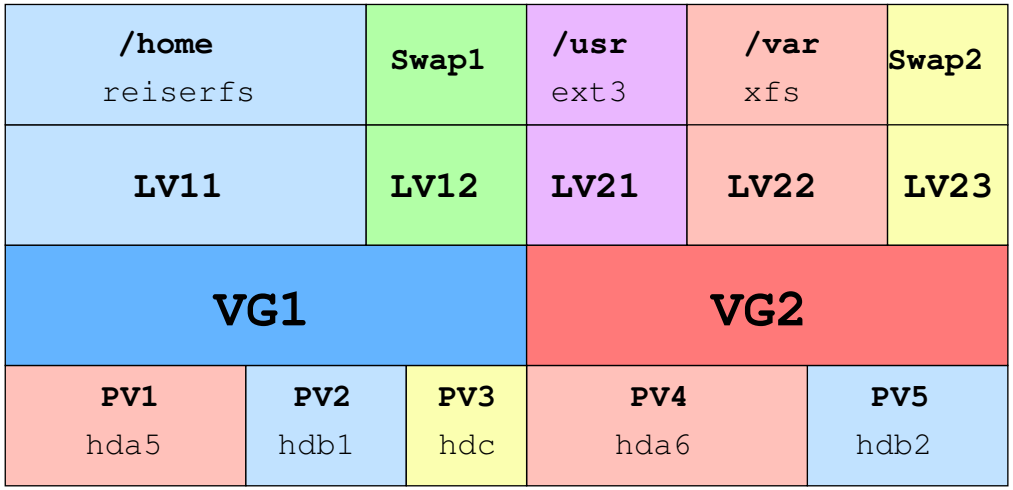
- Device Mapper (Linux 2.6)
  - pilote noyau
  - outils en espace utilisateur : paquet `dmsetup`
- Chiffrement de volume (paquet `dm-crypt`)
- Logical Volume Management (paquet `lvm2`)
  - utilisation plus flexible de l'espace disque
  - IBM AIX (1986-89), HP-UX, Linux 2.2 (1998)
  - Linux 2.6 : lvm2 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 standardisé (POSIX)

Une normalisation POSIX

- Inodes : création de liens durs
- Métadonnées standard
  - horodatage : atime, ctime, mtime
  - permissions POSIX
  - propriétaires : utilisateur et groupe
  - type de fichier
- Rappel : 7 types d'entrées de répertoires (direntries)
  - (f) fichier régulier
  - (d) répertoire
  - (l) lien symbolique
  - (b) périphérique blocs
  - (c) périphérique caractères
  - (p) pipe nommé (FIFO) (IPC)
  - (s) socket (IPC)

Systèmes de fichiers avancés : un aperçu

- Journalisation : ext3, ext4, XFS...
- Métadonnées étendues : attributs libres, ACL...
- Instantanés (snapshots) niveau bloc (LVM) ou fs (ZFS...)
- Verrous
- Compression transparente
- Détection et correction d'erreurs
- Internes : allocation, structure de données
- Algorithme dépendant du périphérique (disque, SSD)

Quelques fonctionnalités avancées ext-2/3/4

Quelques fonctionnalités avancées

- Attributs spécifiques extNfs
- Quotas disque
- Attributs étendus
- ACL (Access Control Lists)

Mise en contexte

- Des fonctionnalités "de niche"
- Exploitées par des applications "haut niveau"
- Pas assez connues des administrateurs
- Biais d'attaque et difficultés de diagnostic

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 : clé=valeur
  ex. `user.creator = "John Doe"`
- Origine XFS, porté sur ext2/3/4 par SGI
- 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 - mise en place

Mise en place

- paquet `attr`
- option noyau (par défaut) CONFIG_EXT2_FS_XATTR=y

Configuration

- `/etc/fstab` : + option `user_xattr`
- `mount -o remount /dev/hdXN`

Documentation

- manpages : attr(5), setfattr(1), getfattr(1)

Les attributs étendus - utilisation

Fixer des attributs

- `setfattr --name="user.lang" --value="fr" fichier`
- `setfattr -n user.src -v www.april.org fichier`

Lire des attributs

- `getfattr -d fichier`
- `getfattr -m <motif> --only-value fichier`

Interactions autres applications

- GNU `tar` les ignore : alternative `star`
- `find` : sur Solaris (SUN) seulement
- `Chrome/Chromium` renseigne `user.xdg.origin.url`

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 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*

Environnement utilisateur (graphique...)

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

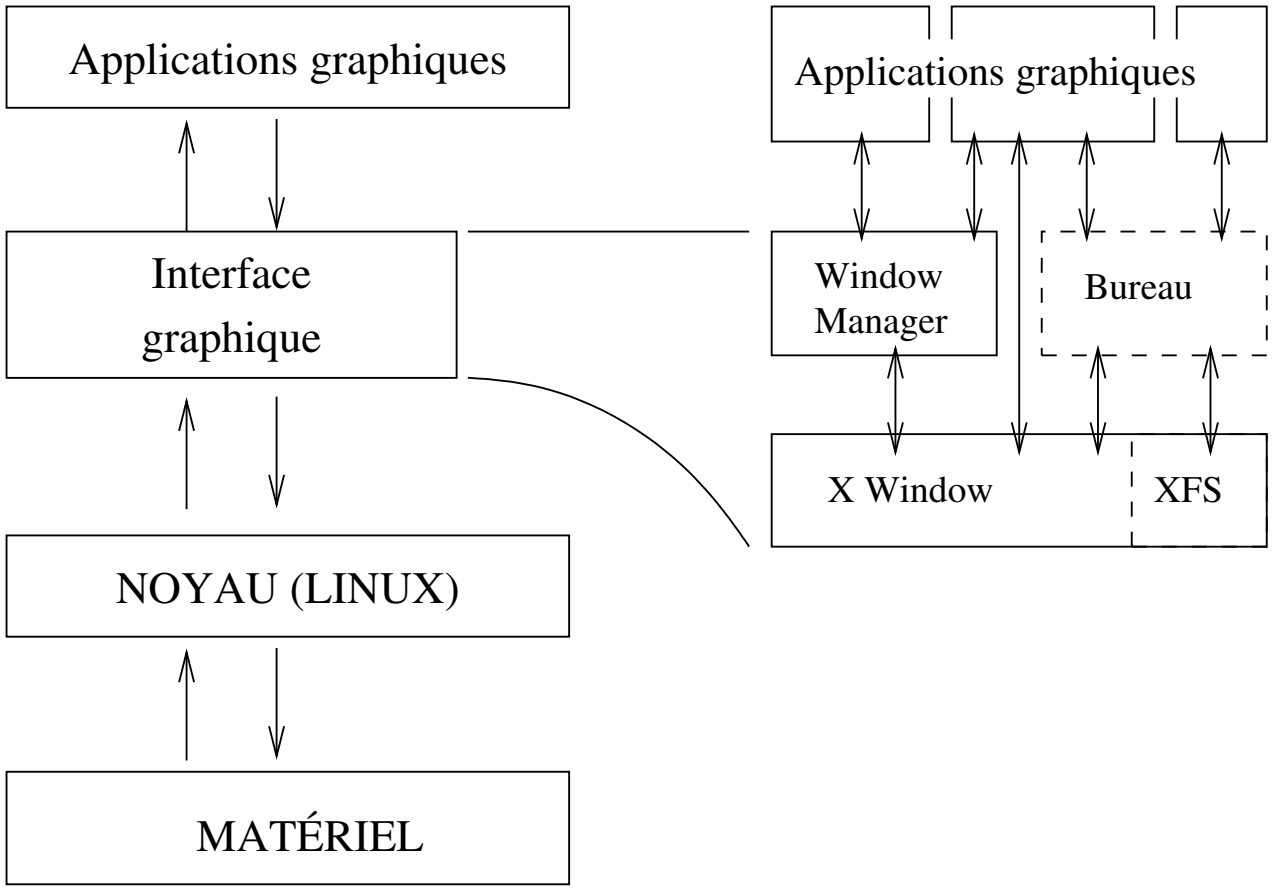
- fichier : `/etc/timezone`
- commandes : `dpkg-reconfigure tzdata`

Configuration du clavier

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

X11
(X Window System)

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 réseau

Architecture TCP/IP

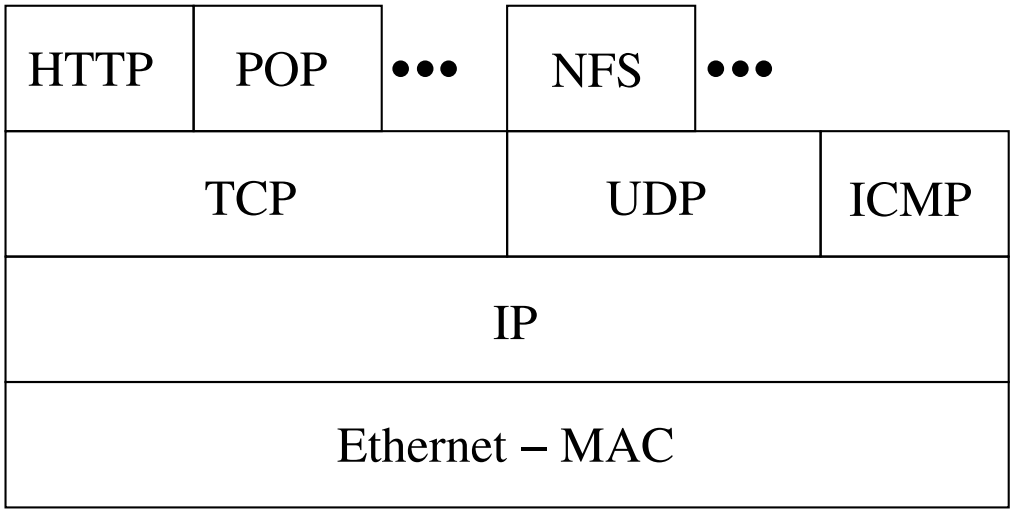
109.1

Un modèle par couches

ernet réseau local Ethernet-MAC

IP l'adressage Internet

TCP le transport



Architecture TCP/IP

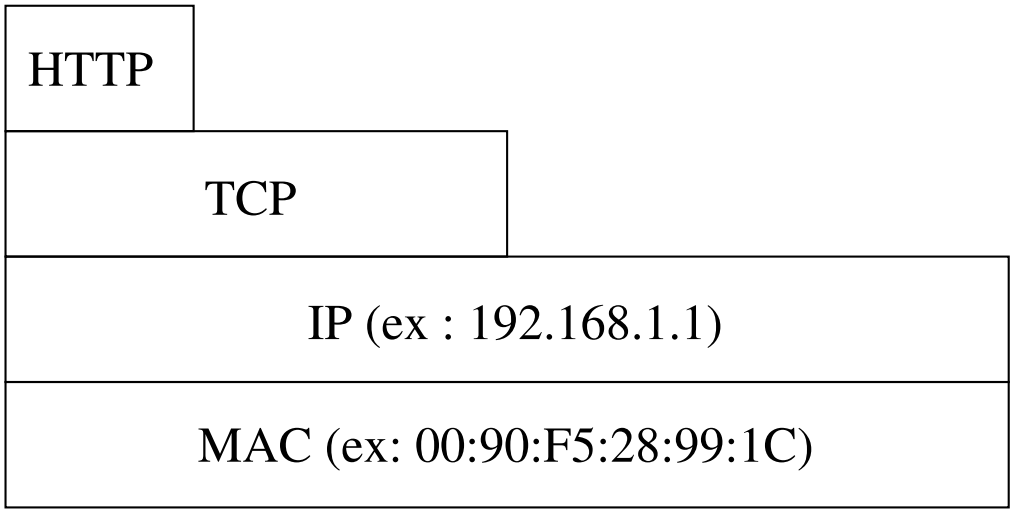
109.1

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 (pré-systemd) *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 (pré-systemd) *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
  - $\underbrace{2001 : 0db8 : 3c4d}_{\text{préfixe global}} : \underbrace{0015}_{\text{sous-réseau}} : \underbrace{0000 : 0000 : abcd : ef12}_{\text{interface}}$

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

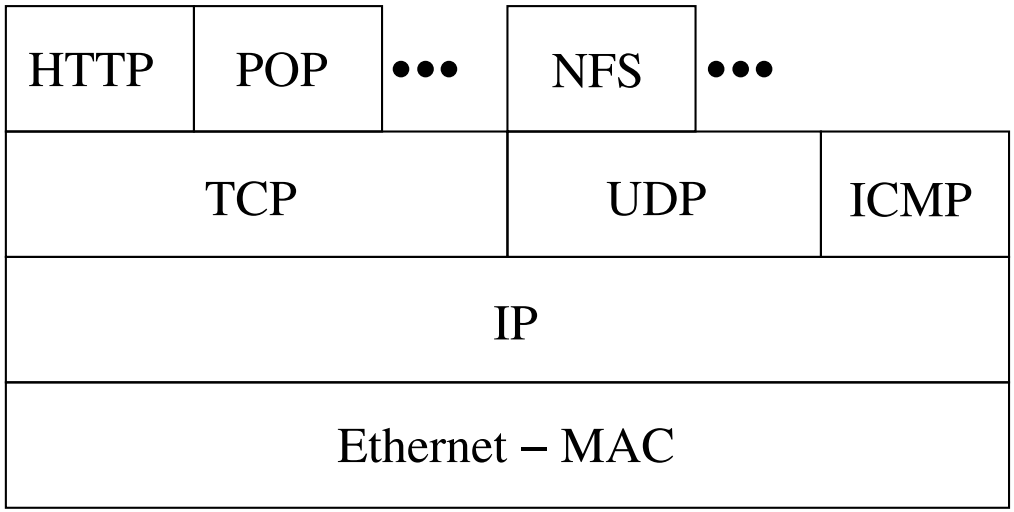
- `whois <objet>` parmi
  - domaine DNS
  - serveur de noms (NS)
  - système autonome (ex. AS12322)
  - adresse IP $\longrightarrow$ AS
  - ... (18 types d'objet)

- 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

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 -l` et `w`
- 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:/home/user/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`.
  - utilisation de `rsync` sur `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

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

Services réseau légers

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

- Bacula : disques + bandes, ligne de commande + GUI
- 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 : Subversion (svn), Mercurial (hg), Git

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

Network Time Protocol

- Idée : une référence de temps distribuée
  - horloge précise (p/r temps universel) : $\approx$ 10 ms
  - horloge uniforme sur le réseau local : $\approx$ 0.2 ms
- Implémentation
  - Protocole UDP, port 123
  - Démon en espace utilisateur (ntpd)
  - + Fonctionnalité noyau : PLL

NTP en pratique

- 3 paquets : `ntpdate`, `ntp`, `ntp-doc`
- `ntpdate <serveur>` : synchronisation isolée
- `ntp` client : synchronisation continue
  - sur un serveur de temps
  - `/etc/ntp.conf` (+ `/etc/default/ntp`)
- `ntp` serveur : 2 modes
  - *push* : broadcast des mises à jour
  - *pull* : autorisation d'accès