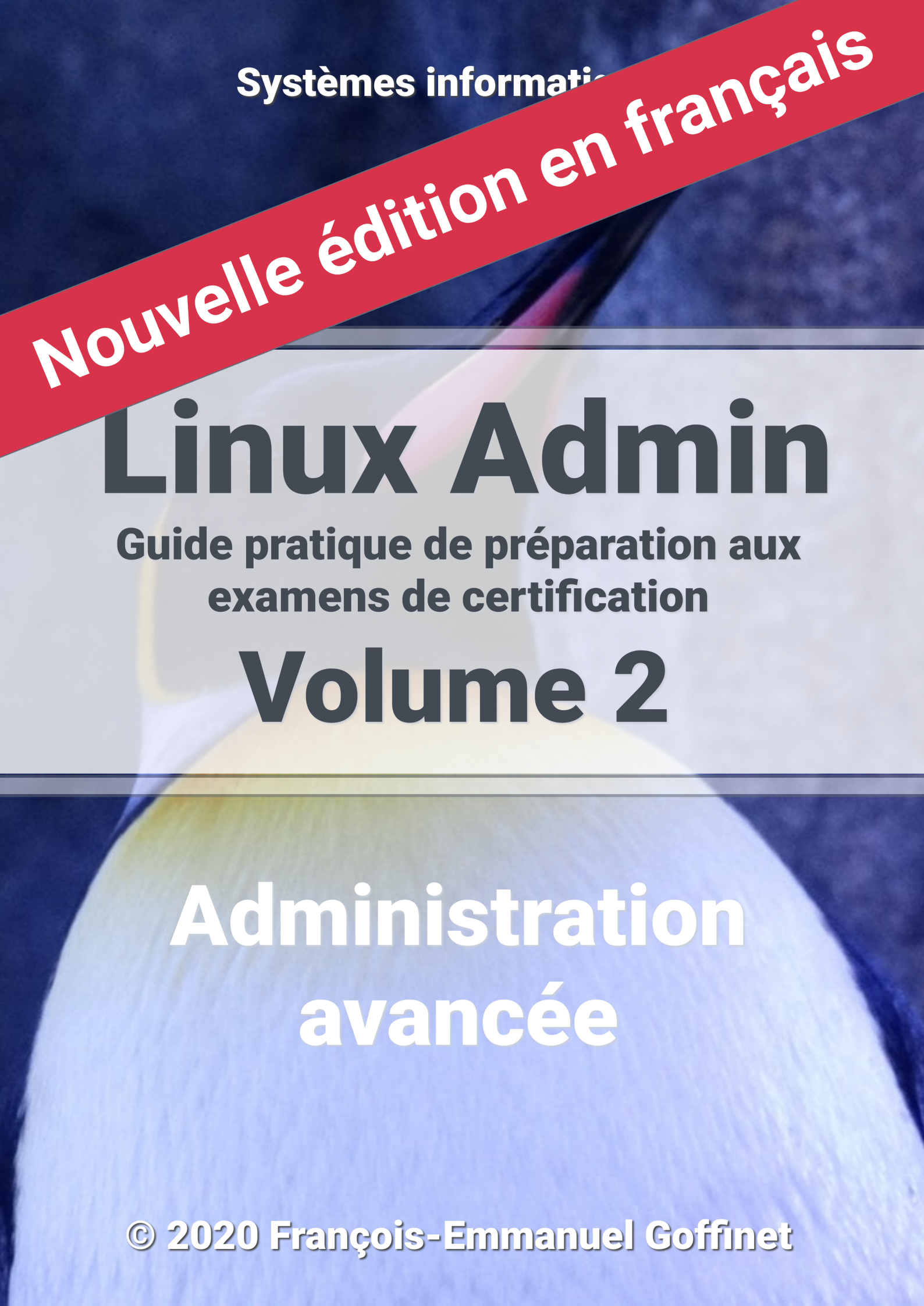

Systèmes informati
Nouvelle édition en français
Linux Admin
Guide pratique de préparation aux examens de certification
Volume 2
Administration avancée
© 2020 François-Emmanuel Goffinet

Linux Administration Volume 2

Linux Administration Avancée. Guide pratique de préparation aux examens de certification LPIC 1, LPIC 2, Linux Essentials, RHCSA et LFCS. Processus et Démarrage. Installation de logiciels. Scripts Shell. Virtualisation KVM. Disques et Stockage LVM. Configuration du réseau.

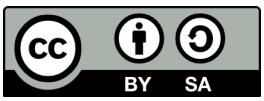
François-Emmanuel Goffinet

Ce livre est en vente à http://leanpub.com/linux-administration-volume-2

Version publiée le 2021-09-06

Leanpub

Ce livre est publié par Leanpub. Leanpub permet aux auteurs et aux éditeurs de bénéficier du Lean Publishing. Lean Publishing consiste à publier à l'aide d'outils très simples de nombreuses itérations d'un livre électronique en cours de rédaction, d'obtenir des retours et commentaires des lecteurs afin d'améliorer le livre.



Aussi par François-Emmanuel Goffinet

Cisco CCNA 200-301 Volume 1

Cisco CCNA 200-301 Volume 2

Cisco CCNA 200-301 Volume 3

Cisco CCNA 200-301 Volume 4

Linux Administration Volume 1

Linux Administration Volume 3

Linux Administration Volume 4

Protocole SIP

Table des matières

Avertissement

Droits

Ce document s'inspire de près ou de loin de toute une série d'autres qui sont soumis la plupart du temps aux mêmes droits. Les sources citées ou reprises sont présentes sous format d'URI dans le code source. J'espère que les auteurs concernés se satisferont de cette exposition. Les marques citées ont été déposées par leurs propriétaires.

Dédicace

À S., amoureusement et à R., tendrement.

Remerciements

Merci aux milliers de visiteurs quotidiens du site linux.goffinet.org.

Merci aux centres de formation et aux écoles qui m'accordent leur confiance et qui me permettent de rencontrer mon public en personne.

Merci à Linux Torvalds qui mérite une reconnaissance universelle pour le don de son oeuvre à l'humanité.

Avant-Propos

François-Emmanuel Goffinet est formateur IT et enseignant depuis 2002 en Belgique et en France. Outre Cisco CCNA, il couvre de nombreux domaines des infrastructures informatiques, du réseau à la virtualisation des systèmes, du nuage à la programmation d'infrastructures hétérogènes en ce y compris DevOps, Docker, K8s, chez AWS, GCP ou Azure, etc. avec une forte préférence et un profond respect pour l'Open Source, notamment pour Linux.

On trouvera ici un des résultats d'un projet d'autopublication en mode *agile* plus large lié au site web linux.goffinet.org.

Ce document fait partie d'un guide de formation en français sur les pratiques sécurisées d'administration du système d'exploitation (OS) GNU/Linux. Le propos invite progressivement à déployer les technologies de virtualisation, à procéder à des tâches d'automation / automatisation via des scripts, à déployer les services traditionnels tels que des services Web ou d'infrastructure, voire plus spécifiques en ToIP / VoIP / UC ou même IaaS.

Le document comprend de nombreux scripts et exemples. Aussi, il traite les sujets sur les distributions basées RHEL (Centos et dérivés) et Debian Stable (Ubuntu et autres dérivés).

Le document vise à atteindre un double objectif :

- Maintenir de manière durable un cours transversal, réutilisable librement, ouvert et actualisé sur les systèmes fonctionnant sous GNU/Linux.
- Aligner les contenus et les pratiques décrites dans les programmes des certifications LPI (Linux Profesional Institute), RH (Red Hat) et LFS (Linux Foundation Software), etc.

Orientation pédagogique

Ce document oriente le contenu sur :

- La virtualisation, les technologies en nuage (*cloud*)
- L'automatisation par la rédaction de code (scripts)
- Les pratiques de sécurité

Public cible du document

Ce document s'adresse à tous les professionnels de l'informatique bien sûr mais aussi des services et de l'industrie pour lesquels l'ère numérique a modifié les pratiques de travail.

Du bon usage du support

Ce support évolue constamment selon l'épreuve du temps et des retours d'expérience. Il est toujours préférable de se référer à la dernière version en ligne sur https ://linux.goffinet.org.

Il se lit ou s'expose en face d'une **console Linux**, dans une machine virtuelle par exemple. Les interfaces graphiques des logiciels seront laissées à l'appréciation des utilisateurs.

Pour obtenir de meilleurs résultats d'apprentissage, notamment en classe de formation, il est conseillé d'utiliser une **installation native**, avec une ligne de commande ou un *shell* à disposition.

Enfin, ce document n'étant qu'un support de cours, il sera nécessaire de visiter les références et les liens fournis ainsi que les sites officiels et leurs pages de documentation qui restent dans la plupart des cas librement disponibles.

Distributions de référence

On conseillera quelques distributions Linux de référence avant d'entamer des distributions spécialisées ou spécifiques.

1. Centos / (RHEL) / Fedora
2. Ubuntu 20.04 LTS Focal / Debian Stable

Matériel nécessaire

Un ordinateur individuel récent connecté au réseau local (et à l'Internet) est nécessaire. Dans une classe de formation, la meilleure expérience est d'installer une distribution Linux native et d'utiliser des outils de virtualisation tels que *libvirt* et *qemu/KVM* pour réaliser des exercices avancés.

Introduction

Ce second volume Linux Administration avancée s'aligne sur les objectifs Linux Essentials, LPIC 1, LPIC 201, RHCSA et LFCS.

Il peut occuper une activité intellectuelle de 16 à 35 heures, voir plus.

L'objectif opérationnel est de maîtriser les aspects avancés de la gestion des processus et du démarrage, des espaces de stockage, du réseau et de la virtualisation notamment grâce à des scripts Shell.

Une première partie intitulée "Processus et démarrage" évoque en premier lieu le noyau Linux et les services qu'il rend au niveau matériel et avec des modules. Dans un second chapitre, on aborde le processus de démarrage et de password recovery d'un système Linux. Un troisième chapitre s'intéresse à la manipulation des processus Linux. Et enfin, on appréciera dans un dernier propos l'utilité d'un logiciel de console virtuelle comme screen.

La seconde partie intitulée "Installation de logiciel" aborde le vaste sujet de la gestion des logiciels sous Linux. Un premier chapitre explique le principe des installateurs principaux des distributions basées Red Hat et Debian sans éluder les autres systèmes. Un second chapitre s'attache au processus d'une compilation par les sources de logiciels sous Linux. Enfin, le propos se termine sur la mise en place de dépôt de paquetages et les installations automatiques.

La troisième partie est une initiation au scripting Shell. On y aborde les notions de base des scripts Bash, les structures conditionnelles, les boucles, des concepts avancés sur les variables, ainsi que des modèles, des figures et des exemples de scripts.

Une quatrième partie s'intéresse à la virtualisation Linux avec Qemu/KVM et la librairie Libvirt et différents outils de gestion Open Source. Un projet de scripts Bash prêt à l'emploi pour le déploiement et l'orchestration de machines virtuelles est proposé comme illustration.

Une cinquième partie sur les disques et le stockage LVM porte sur les différents types de systèmes de fichiers, le formatage, le schéma de partitionnement, les tables MBR et GPT, les points de montage, la mémoire SWAP, la manipulation des disques RAID 1, RAID 5 et RAID 10. On trouvera aussi un développement sur le stockage LVM sur un plan théorique et pratique.

Une sixième partie sur la configuration réseau termine l'ouvrage. On trouvera en premier lieu une introduction à TCP/IP, ensuite une synthèse rapide des commandes de diagnostic à retenir. Les trois chapitres qui suivent étudient la gestion du réseau sous Linux avec Network Manager, avec la librairie Iproute2 et Netplan. Enfin, cette partie se termine par la revue de différents outils réseau sous Linux.

Première partie Processus et démarrage

Objectifs de certification

Linux Essentials

- *Sujet 4 : Le système d'exploitation Linux*
  - 4.2 Compréhension du matériel informatique (valeur : 2)

RHCSA EX200

- **3. Utiliser des systèmes en cours d'exécution**
  - 3.1. Démarrer, redémarrer et éteindre un système normalement
  - 3.2. Démarrer des systèmes dans différentes cibles manuellement
  - 3.3. Interrompre le processus de démarrage afin d'obtenir l'accès à un système
  - 3.4. Identifier les processus exigeants en processeur/mémoire, arrêter des processus
  - 3.5. Adapter la planification du processus
  - 3.6. Gérer les profils de tuning
- **6. Déployer, configurer et gérer des systèmes**
  - 6.2. Démarrer et arrêter des services, et configurer des services pour qu'ils se lancent automatiquement au démarrage
  - 6.3. Configurer des systèmes pour démarrer automatiquement dans une cible spécifique
  - 6.6. Modifier le chargeur de démarrage du système

RHCE EX300

- 1. **Configuration et gestion de systèmes**
  - 1.7. Générer et livrer des rapports sur l'utilisation du système (processeur, mémoire, disque et réseau)

LPIC 1

- *Sujet 101 : Architecture système*
  - 101.1 Détermination et configuration des paramètres du matériel
  - 101.2 Démarrage du système
  - 101.3 Changement de niveaux d'exécution / des cibles de démarrage de systemd et arrêt ou redémarrage du système
- *Sujet 102 : Installation de Linux et gestion de paquetages*
  - 102.2 Installation d'un gestionnaire d'amorçage
  - 102.3 Gestion des bibliothèques partagées
- *Sujet 103 : Commandes GNU et Unix*
  - 103.4 Utilisation des flux, des tubes et des redirections
  - 103.5 Création, contrôle et interruption des processus
  - 103.6 Modification des priorités des processus

LPIC 2

- *Sujet 200 : Planification des ressources*
  - 200.1 Mesure de l'utilisation des ressources et résolution de problèmes (valeur : 6)
  - 200.2 Prévision des besoins en ressources (valeur : 2)
- *Sujet 201 : le noyau Linux*
  - 201.1 Composants du noyau (valeur : 2)
  - 201.2 Compilation du noyau (valeur : 3)
  - 201.3 Gestion du noyau à chaud et résolution de problèmes (valeur : 4)
- *Sujet 202 : Démarrage du système*
  - 202.1 Personnalisation du démarrage système (valeur :3)
  - 202.2 Récupération du système (valeur : 4)
  - 202.3 Chargeurs d'amorçage alternatifs (valeur : 2)

Introduction

Cette partie évoque en premier lieu le noyau Linux et les services qu'il rend au niveau matériel et avec des modules. Dans un second chapitre, on aborde le processus de démarrage et de password recovery d'un système Linux. Un troisième chapitre s'intéresse à la manipulation des processus Linux. Et enfin, on appréciera dans un dernier propos l'utilité d'un logiciel de console virtuelle comme screen.

Références

- https ://fr.wikipedia.org/wiki/Sysfs
- https ://fr.wikibooks.org/wiki/Le_syst%C3%A8me_d%27exploitation_GNU-Linux/Le_noyau_Linux_et_les_-modules
- [https ://fr.wikibooks.org/wiki/Le_syst%C3%A8me_d%27exploitation_GNU-Linux/Les_p%C3%A9riph%C3%A9riques/*de*
  fr.wikibooks.org/wiki/Le_syst%C3%A8me_d%27exploitation_GNU-Linux/Les_p%C3%A9riph%C3%A9riques/dev)
- https ://www.thegeekstuff.com/2010/11/linux-proc-file-system/
- https ://en.wikipedia.org/wiki/Procfs
- https ://www.tldp.org/LDP/Linux-Filesystem-Hierarchy/html/proc.html
- https ://fr.wikipedia.org/wiki/Initrd
- https ://en.wikipedia.org/wiki/System.map
- https ://en.wikipedia.org/wiki/Linux_startup_process
- https ://0pointer.de/blog/projects/systemd-docs.html
- https ://fedoraproject.org/wiki/Systemd
- https ://en.wikipedia.org/wiki/Init
- https ://fr.wikipedia.org/wiki/Systemd
- https ://www.tecmint.com/systemd-replaces-init-in-linux/
- https ://fedoraproject.org/wiki/SysVinit_to_Systemd_Cheatsheet
- https ://www.freedesktop.org/wiki/Software/systemd/
- https ://wiki.archlinux.org/index.php/SysVinit

1. Noyau Linux

Ce chapitre est une introduction au noyau Linux. On expliquera ici comment interroger et configurer un noyau Linux courant.

1. Noyau Linux

Source : https ://fr.wikipedia.org/wiki/Noyau_Linux

1.1. Généralités

Le noyau Linux est un noyau de système d'exploitation de type UNIX. Le noyau Linux est un logiciel libre développé essentiellement en langage C par des milliers de bénévoles et salariés communiquant par Internet.

Le noyau est le cœur du système, c'est lui qui s'occupe de fournir aux logiciels une interface pour utiliser le matériel. Le noyau Linux a été créé en 1991 par Linus Torvalds pour les compatibles PC construits sur l'architecture processeur x86. Depuis, il a été porté sur nombre d'architectures dont m68k, PowerPC, StrongARM, Alpha, SPARC, MIPS, etc. Il s'utilise dans une très large gamme de matériel, des systèmes embarqués aux superordinateurs, en passant par les ordinateurs personnels.

Ses caractéristiques principales sont d'être multitâche et multi-utilisateur. Il respecte les normes POSIX ce qui en fait un digne héritier des systèmes UNIX. Au départ, le noyau a été conçu pour être monolithique. Ce choix technique fut l'occasion de débats enflammés entre Andrew S. Tanenbaum, professeur à l'université libre d'Amsterdam qui avait développé Minix, et Linus Torvalds. Andrew Tanenbaum arguant que les noyaux modernes se devaient d'être des micro-noyaux et Linus répondant que les performances des micronoyaux n'étaient pas bonnes. Depuis sa version 2.0, le noyau, bien que n'étant pas un micro-noyau, est modulaire, c'est-à-dire que certaines fonctionnalités peuvent être ajoutées ou enlevées du noyau à la volée (en cours d'utilisation).

1.2. Développement du noyau Linux

Si au début de son histoire le développement du noyau Linux était assuré par des développeurs bénévoles, les principaux contributeurs sont aujourd'hui un ensemble d'entreprises, souvent concurrentes, comme Red Hat, Novell, IBM ou Intel.

La licence du noyau Linux est la licence publique générale GNU dans sa version 2. Cette licence est libre, ce qui permet d'utiliser, copier et modifier le code source selon ses envies ou ses besoins. Ainsi, quiconque a les connaissances nécessaires peut participer aux tests et à l'évolution du noyau.

Linus Torvalds, créateur du noyau Linux, est le mainteneur officiel depuis le début en 1991. Il est une sorte de « dictateur bienveillant », l'autorité en termes de choix techniques et organisationnels. Les différentes versions du noyau publiées par Linus Torvalds s'appellent « mainline » ou « vanilla » en anglais. Ce sont les noyaux vanilla qui sont intégrés par les distributeurs, avec parfois l'addition de quelques patchs de sécurité, de corrections de bogue ou d'optimisations.

Linus Torvalds a apporté un changement radical dans la façon dont les systèmes d'exploitation sont développés, en utilisant pleinement la puissance du réseau Internet.

Le processus de développement de Linux est public sur Internet : les sources du noyau y sont visibles par tous, les modifications de ces sources sont publiées et revues sur Internet et sont également visibles de tous. Un cycle de développement incrémental et rapide a été adopté depuis le début (aujourd'hui une nouvelle version est publiée

toutes les 9 semaines environ), qui a permis de construire autour de Linux et d'Internet par couches successives une communauté dynamique composée de développeurs, de sociétés et d'utilisateurs.

Les numéros de version du noyau sont composés de trois nombres : le premier est le numéro majeur, le second le numéro mineur. Avant l'apparition des versions 2.6.x, les numéros mineurs pairs indiquaient une version stable et les numéros mineurs impairs une version de développement. Ainsi, les versions 2.2, 2.4 sont stables, les versions 2.3 et 2.5 sont des versions de développement. Cependant, depuis la version 2.6 du noyau, ce modèle de numérotation stable/développement a été abandonné et il n'y a donc plus de signification particulière aux numéros mineurs pairs ou impairs. Le troisième nombre indique une révision, ce qui correspond à des corrections de bogues, de sécurité ou un ajout de fonctionnalité, par exemple 2.2.26, 2.4.30 ou 2.6.11. Le passage à la version 3.0 fut décidé par Linus Torvalds à l'occasion des 20 ans du noyau Linux, même si la véritable raison fut plutôt arbitraire. La dernière version stable en mai 2017 est 4.11.1.

Cette page https ://fr.wikipedia.org/wiki/Noyau_Linux#Chronologie donne une idée de l'évolution des intégrations au noyau.

1.3. Version courante du noyau

La commande `uname` permet de connaître la version courante du noyau, mais aussi le type d'architecture et le nom de l'ordinateur.

Par exemple, sur une Centos 7 :

```
uname -a
Linux srv.linuxlab.be 3.10.0-327.18.2.el7.x86_64 #1 SMP Thu May 12 11:03:55 UTC 2016 x86_64\
 x86_64 x86_64 GNU/Linux
uname -r
3.10.0-327.18.2.el7.x86_64
uname -m
x86_64
uname -n
srv.linuxlab.be
uname -p
x86_64
uname -s
Linux
uname -i
x86_64
uname -o
GNU/Linux
uname -v
#1 SMP Thu May 12 11:03:55 UTC 2016
```

Le fichier `/proc/cmdline` informe notamment du noyau utilisé par le système.

```
cat /proc/cmdline
BOOT_IMAGE=/boot/vmlinuz-3.10.0-327.18.2.el7.x86_64 root=/dev/md1 ro net.ifnames=0 rd.md.uu\
id=c2dd5ffb:ee7dff79:a4d2adc2:26fd5302 kvm-intel.nested=1
```

1.4. Messages du noyau

Le noyau écrit ses événements dans et via dmesg qui sont consultés après une nouvelle installation.

```
# head /var/log/messages
Aug 26 11:01:32 sb1 rsyslogd: [origin software="rsyslogd" swVersion="7.4.7" x-pid="594" x-i\
nfo="https://www.rsyslog.com"] start
Aug 26 11:01:25 sb1 journal: Runtime journal is using 6.2M (max allowed 49.6M, trying to le\
ave 74.4M free of 490.3M available  current limit 49.6M).
Aug 26 11:01:25 sb1 journal: Runtime journal is using 6.2M (max allowed 49.6M, trying to le\
ave 74.4M free of 490.3M available  current limit 49.6M).
Aug 26 11:01:25 sb1 kernel: Initializing cgroup subsys cpuset
Aug 26 11:01:25 sb1 kernel: Initializing cgroup subsys cpu
Aug 26 11:01:25 sb1 kernel: Initializing cgroup subsys cpuacct
Aug 26 11:01:25 sb1 kernel: Linux version 3.10.0-327.el7.x86_64 (builder@kbuilder.dev.cento\
s.org) (gcc version 4.8.3 20140911 (Red Hat 4.8.3-9) (GCC) ) #1 SMP Thu Nov 19 22:10:57 UTC\
 2015
Aug 26 11:01:25 sb1 kernel: Command line: BOOT_IMAGE=/vmlinuz-3.10.0-327.el7.x86_64 root=/d\
ev/mapper/centos_tmp--30faa133-root ro crashkernel=auto rd.lvm.lv=centos_tmp-30faa133/root \
rd.lvm.lv=centos_tmp-30faa133/swap console=ttyS0,115200n8 LANG=en_US.UTF-8
Aug 26 11:01:25 sb1 kernel: e820: BIOS-provided physical RAM map:
Aug 26 11:01:25 sb1 kernel: BIOS-e820: [mem 0x0000000000000000-0x000000000009fbff] usable
```

`dmesg` (pour l'anglais "display message", "afficher message" en français) est une commande sur les systèmes d'exploitation de type Unix qui affiche la mémoire tampon de message du noyau. Elle permet de vérifier le comportement du noyau, notamment le sort réservé aux pilotes de périphérique (modules du noyau). On trouvera le contenu dans le fichier `/var/log/dmesg` ou encore dans `/var/log/kern.log` (Debian/Ubuntu) .

```
# dmesg
```

Peut donner :

```
[ 0.000000] Initializing cgroup subsys cpuset
[ 0.000000] Initializing cgroup subsys cpu
[ 0.000000] Initializing cgroup subsys cpuacct
[ 0.000000] Linux version 3.10.0-327.el7.x86_64 (builder@kbuilder.dev.centos.org) (gcc vers\
ion 4.8.3 20140911 (Red Hat 4.8.3-9) (GCC) ) #1 SMP Thu Nov 19 22:10:57 UTC 2015
[ 0.000000] Command line: BOOT_IMAGE=/vmlinuz-3.10.0-327.el7.x86_64 root=/dev/mapper/centos\
_tmp--30faa133-root ro crashkernel=auto rd.lvm.lv=centos_tmp-30faa133/root rd.lvm.lv=centos\
_tmp-30faa133/swap console=ttyS0,115200n8 LANG=en_US.UTF-8
[ 0.000000] e820: BIOS-provided physical RAM map
...
```

Pour une recherche plus précise, par exemple :

```
# dmesg | grep vda
[ 1.938760] vda: vda1 vda2
[ 8.081330] XFS (vda1): Mounting V4 Filesystem
[ 8.343152] XFS (vda1): Ending clean mount
[ 8.345418] SELinux: initialized (dev vda1, type xfs), uses xattr
```

```
# dmesg | grep kvm
[ 0.000000] kvm-clock: Using msrs 4b564d01 and 4b564d00
[ 0.000000] kvm-clock: cpu 0, msr 0:3ff87001, primary cpu clock
[ 0.000000] kvm-clock: using sched offset of 5508623650 cycles
[ 0.000000] kvm-stealtime: cpu 0, msr 3fc0d240
[ 0.746986] Switching to clocksource kvm-clock
[ 1.426438] systemd[1]: Detected virtualization kvm.
```

1.5. Documentation du noyau

Sous Centos/RHEL, l'accès local à la documentation nécessite l'installation du paquet `kernel-doc`. Selon la version du système, les fichiers de documentation seront copiés dans un répertoire du type `/usr/share/doc/kernel-doc-3.10.0/Documer`

Sous Debian/Ubuntu, la documentation accompagne d'emblée le noyau. Elle se situe dans le répertoire `Documentation` du noyau. `/usr/src/linux-headers-$(uname -r)/Documentation`

La documentation Web se trouve sur https ://www.kernel.org/doc.

1.6. Configuration de paramètres du noyau

On peut changer les paramètres du noyau à chaud en écrivant directement les valeurs dans les fichiers adéquats (`/proc/sys/`) ou en utilisant le binaire `sysctl`.

Par exemple, on peut vérifier si le routage IPv4 est activé :

```
# cat /proc/sys/net/ipv4/ip_forward
0
```

Il suffit de placer la valeur à `1` dans ce fichier pour activer le routage :

```
# echo 1 > /proc/sys/net/ipv4/ip_forward
# cat /proc/sys/net/ipv4/ip_forward
1
```

On aurait pu exécuter la même opération avec `sysctl`.

1.7. Sysctl

`sysctl` est le programme qui permet de modifier à chaud les paramètres du noyau.

```
# sysctl --help

Usage:
sysctl [options] [variable[=value] ...]

Options:
-a, --all display all variables
-A alias of -a
-X alias of -a
--deprecated include deprecated parameters to listing
-b, --binary print value without new line
```

```
-e, --ignore ignore unknown variables errors
-N, --names print variable names without values
-n, --values print only values of a variables
-p, --load[=<file>] read values from file
-f alias of -p
--system read values from all system directories
-r, --pattern <expression>
select setting that match expression
-q, --quiet do not echo variable set
-w, --write enable writing a value to variable
-o does nothing
-x does nothing
-d alias of -h

-h, --help display this help and exit
-V, --version output version information and exit

For more details see sysctl(8).
```

- `sysctl -a` affiche toutes les variables avec leur valeur
- `sysctl -n [variable]` affiche valeur d'une variable comme par exemple `sysctl -n net.ipv4.ip_forward`.

Pour modifier un paramètre du noyau avec `sysctl` :

```
# sysctl net.ipv4.ip_forward=1
net.ipv4.ip_forward = 1
```

1.8. Persistence des paramètres du noyau

Enfin, pour rendre ces paramétrages permanents :on peut valoriser ces variables dans le fichiers `/etc/sysctl.conf`.

Ici un exemple Ubuntu 12.04 par défaut :

```
# /etc/sysctl.conf - Configuration file for setting system variables
# See /etc/sysctl.d/ for additional system variables
# See sysctl.conf (5) for information.
#

#kernel.domainname = example.com

# Uncomment the following to stop low-level messages on console
#kernel.printk = 3 4 1 3

##############################################################3
# Functions previously found in netbase
#

# Uncomment the next two lines to enable Spoof protection (reverse-path filter)
# Turn on Source Address Verification in all interfaces to
# prevent some spoofing attacks
#net.ipv4.conf.default.rp_filter=1
```

```
#net.ipv4.conf.all.rp_filter=1

# Uncomment the next line to enable TCP/IP SYN cookies
# See https://lwn.net/Articles/277146/
# Note: This may impact IPv6 TCP sessions too
#net.ipv4.tcp_syncookies=1

# Uncomment the next line to enable packet forwarding for IPv4
#net.ipv4.ip_forward=1

# Uncomment the next line to enable packet forwarding for IPv6
# Enabling this option disables Stateless Address Autoconfiguration
# based on Router Advertisements for this host
#net.ipv6.conf.all.forwarding=1


###################################################################
# Additional settings - these settings can improve the network
# security of the host and prevent against some network attacks
# including spoofing attacks and man in the middle attacks through
# redirection. Some network environments, however, require that these
# settings are disabled so review and enable them as needed.
#
# Do not accept ICMP redirects (prevent MITM attacks)
#net.ipv4.conf.all.accept_redirects = 0
#net.ipv6.conf.all.accept_redirects = 0
# _or_
# Accept ICMP redirects only for gateways listed in our default
# gateway list (enabled by default)
# net.ipv4.conf.all.secure_redirects = 1
#
# Do not send ICMP redirects (we are not a router)
#net.ipv4.conf.all.send_redirects = 0
#
# Do not accept IP source route packets (we are not a router)
#net.ipv4.conf.all.accept_source_route = 0
#net.ipv6.conf.all.accept_source_route = 0
#
# Log Martian Packets
#net.ipv4.conf.all.log_martians = 1
#
```

2. Configuration matérielle

2.1. Le système de fichiers virtuel `/proc`

- `/proc` n'existe pas sur le disque dur, il est fournit dynamiquement par le noyau, d'où le nom de virtuel.
- Il permet de fournir des informations sur ce que voit le noyau.
- En outre pour accéder à certains renseignements il sera nécessaire de monter obligatoirement `/proc` :

```
# mount | grep \/proc
proc on /proc type proc (rw,nosuid,nodev,noexec,relatime)
```

- Les commandes `ps`, `top`, `uptime` (et bien d'autres) utilisent `/proc` pour récupérer des informations du système.

2.2. Informations de bas niveau

```
cat /proc/interrupts
```

- VM : https ://pastebin.com/ZVuh2UK0
- APU1D4 : https ://pastebin.com/wqAAHrN6
- Rpi1 : https ://pastebin.com/ZVuh2UK0

```
cat /proc/dma; cat /proc/ioports
```

- VM : https ://pastebin.com/aEE3thaV

```
cat /proc/devices
```

- VM : https ://pastebin.com/m6dJUiTD

2.3. Information sur les bus

```
# lspci
```

- APU1D4 : https ://pastebin.com/2D1Nzk3U

```
# lsusb -t
```

2.4. Informations CPU, mémoires, RAM, etc.

```
cat /proc/cpuinfo
cat /proc/meminfo
cat /proc/loadavg
cat /proc/partitions
cat /proc/version
cat /proc/mounts
cat /proc/stat
cat /proc/uptime
cat /proc/swaps
```

mais aussi,

```
lscpu
free -m
vmstat -s
```

2.5. Fichier `/proc/$PID`

- `/proc` contient aussi les numéros des processus et les informations les concernant, par exemple sur un processus SSHD :

```
ps aux | grep sshd
root 1204 0.0 0.0 61364 3080 ? Ss 2014 0:00 /usr/sbin/sshd -D
root 25741 0.0 0.1 105632 4264 ? Ss 10:06 0:00 sshd: francois [priv]
francois 25840 0.0 0.0 105632 2168 ? S 10:07 0:02 sshd: francois@pts/5
root 25986 0.0 0.0 11768 916 pts/5 S+ 11:27 0:00 grep --color=auto sshd
```

```
cat /proc/1204/status
Name:        sshd
State:        S (sleeping)
Tgid:        1204
Ngid:        0
Pid:        1204
PPid:        1
...
```

2.6. Périphériques `/dev`

- Linux accède aux périphériques à partir de fichiers situés dans `/dev`.
- Les disques durs :

```
ls -l /dev/sd*
brw-rw---- 1 root disk 8, 0 déc 15 23:40 /dev/sda
brw-rw---- 1 root disk 8, 1 déc 15 23:40 /dev/sda1
brw-rw---- 1 root disk 8, 2 déc 15 23:40 /dev/sda2
brw-rw---- 1 root disk 8, 5 déc 15 23:40 /dev/sda5
brw-rw---- 1 root disk 8, 16 déc 15 23:40 /dev/sdb
```

Où nous avons des fichiers de type block (b) avec un numéro primaire 8 et un numéro secondaire qui identifie la partitions pour le noyau.

- La commande blkid permet d'identifier les périphériques block par leur UUID :

```
blkid
/dev/sda1: UUID="67407b6c-4bbc-4b52-b071-fee802cfbf2b" TYPE="xfs"
/dev/sda2: UUID="2e468ba5-a730-4988-b8e6-3073a048227f" TYPE="swap"
```

2.7. Tous les autres périphériques `/dev`

- Ce dossier contient tous les périphériques matériels comme un lecteur cdrom, une carte son, une carte réseau, etc…
- Il contient également les pseudo-périphériques. Quelques exemples :
- `/dev/zero` génére des zéros
- `/dev/random` génère de l'aléatoire
- `/dev/null` constitue un trou noir à octets, et notamment utilisé pour se débarrasser des fichiers et des affichages
- `/dev/loop0` permet de créer de faux périphériques de type block (stockage) à partir de fichiers créés avec la commande dd
- Si on liste le contenu de /dev :

```
# ls -l /dev | more
```

Exercices pratiques : se connecter en console sur un routeur, un commutateur, un ordinateur embarqué. Indications : commande `screen`, vitesse, `/dev/ttyS0`, `/dev/ttyUSB0`. Comment connecter deux ordinateurs Linux via leur port série ou USB ?

2.8. Sysfs

- Sysfs est un système de fichiers virtuel introduit par le noyau Linux 2.6. Sysfs permet d'exporter depuis l'espace noyau vers l'espace utilisateur des informations sur les périphériques du système et leurs pilotes, et est également utilisé pour configurer certaines fonctionnalités du noyau.

3. Modules et fichiers du noyau

3.1. Modules du noyau

- Un module du noyau est un pilote de périphérique utilisé par le noyau pour utiliser le matériel et les logiciels.
- `lsmod` permet de voir les modules chargés dans le noyau :

```
lsmod
```

- Emplacements des modules du noyau (Centos 7 noyau `3.10.0-327.*el7.x86_64`)

```
ls /lib/modules
3.10.0-327.18.2.el7.x86_64 3.10.0-327.el7.x86_64
```

```
ls -l /lib/modules/3.10.0-327.18.2.el7.x86_64/
total 2704
lrwxrwxrwx. 1 root root 43 11 jun 16:53 build -> /usr/src/kernels/3.10.0-327.18.2.el7.x86_64
drwxr-xr-x. 2 root root 6 12 mai 13:15 extra
drwxr-xr-x. 11 root root 4096 11 jun 16:53 kernel
-rw-r--r--. 1 root root 706371 11 jun 16:53 modules.alias
-rw-r--r--. 1 root root 682782 11 jun 16:53 modules.alias.bin
-rw-r--r--. 1 root root 1288 12 mai 13:16 modules.block
-rw-r--r--. 1 root root 5995 12 mai 13:13 modules.builtin
-rw-r--r--. 1 root root 7744 11 jun 16:53 modules.builtin.bin
-rw-r--r--. 1 root root 218218 11 jun 16:53 modules.dep
-rw-r--r--. 1 root root 316220 11 jun 16:53 modules.dep.bin
-rw-r--r--. 1 root root 339 11 jun 16:53 modules.devname
-rw-r--r--. 1 root root 108 12 mai 13:16 modules.drm
-rw-r--r--. 1 root root 100 12 mai 13:16 modules.modesetting
-rw-r--r--. 1 root root 1522 12 mai 13:16 modules.networking
-rw-r--r--. 1 root root 84666 12 mai 13:13 modules.order
-rw-r--r--. 1 root root 89 11 jun 16:53 modules.softdep
-rw-r--r--. 1 root root 311931 11 jun 16:53 modules.symbols
-rw-r--r--. 1 root root 387108 11 jun 16:53 modules.symbols.bin
lrwxrwxrwx. 1 root root 5 11 jun 16:53 source -> build
drwxr-xr-x. 2 root root 6 12 mai 13:15 updates
drwxr-xr-x. 2 root root 91 11 jun 16:53 vdso
drwxr-xr-x. 2 root root 6 12 mai 13:15 weak-updates


ls /lib/modules/3.10.0-327.18.2.el7.x86_64/kernel/fs
binfmt_misc.ko ceph dlm fat gfs2 lockd nfs_common overlayfs udf
btrfs cifs exofs fscache isofs mbcache.ko nfsd pstore xfs
cachefiles cramfs ext4 fuse jbd2 nfs nls squashfs
```

- Dépendances des modules entre eux :

```
depmod 3.10.0-327.18.2.el7.x86_64


head /lib/modules/3.10.0-327.18.2.el7.x86_64/modules.dep
kernel/arch/x86/kernel/cpu/mcheck/mce-inject.ko:
kernel/arch/x86/kernel/test_nx.ko:
kernel/arch/x86/crypto/ablk_helper.ko: kernel/crypto/cryptd.ko
kernel/arch/x86/crypto/glue_helper.ko:
kernel/arch/x86/crypto/camellia-x86_64.ko: kernel/crypto/xts.ko kernel/crypto/lrw.ko kernel\
/crypto/gf128mul.ko kernel/arch/x86/crypto/glue_helper.ko
kernel/arch/x86/crypto/blowfish-x86_64.ko: kernel/crypto/blowfish_common.ko
kernel/arch/x86/crypto/twofish-x86_64.ko: kernel/crypto/twofish_common.ko
kernel/arch/x86/crypto/twofish-x86_64-3way.ko: kernel/arch/x86/crypto/twofish-x86_64.ko ker\
nel/crypto/twofish_common.ko kernel/crypto/xts.ko kernel/crypto/lrw.ko kernel/crypto/gf128m\
ul.ko kernel/arch/x86/crypto/glue_helper.ko
kernel/arch/x86/crypto/salsa20-x86_64.ko:
kernel/arch/x86/crypto/serpent-sse2-x86_64.ko: kernel/crypto/xts.ko kernel/crypto/serpent_g\
eneric.ko kernel/crypto/lrw.ko kernel/crypto/gf128mul.ko kernel/arch/x86/crypto/glue_helper\
.ko kernel/arch/x86/crypto/ablk_helper.ko kernel/crypto/cryptd.ko
```

3.2. Charger / décharger un module

On peut charger un pilote de périphérique. Toute une série sont déjà pour les cartes réseau dans `/lib/modules/$(uname -r)/kernel/drivers/net` :

```
ls /lib/modules/$(uname -r)/kernel/drivers/net
appletalk dummy.ko geneve.ko ipvlan mii.ko plip sungem_phy.ko vrf.ko xen-netback
arcnet eql.ko hamradio irda netconsole.ko ppp team vxlan.ko
bonding ethernet hyperv macvlan.ko nlmon.ko rionet.ko usb wan
caif fddi ieee802154 macvtap.ko ntb_netdev.ko sb1000.ko veth.ko wimax
can fjes ifb.ko mdio.ko phy slip vmxnet3 wireless
```

Sous Ubuntu 14.04 dans une machine virtuelle VMWare, on peut par exemple tenter de charger le pilote d'une carte vmxnet3 :

```
$ sudo insmod /proc/lib/modules/4.4.0-31-generic/kernel/drivers/net/vmxnet3/vmxnet3.ko


$ modprobe vmxnet3


$ lsmod | grep vmxnet3
vmxnet3 57344 0


$ sudo rmmod vmxnet3
```

- On peut charger ou décharger un module du noyau avec `modprobe` au lieu de la commande `insmod` :

```
# modprobe msdos
# lsmod | grep msdos
msdos 17332 0
fat 65913 1 msdos
# rmmod msdos
# lsmod | grep msdos
```

3.3. UDEV

3.4. Fichiers du noyau

Les fichiers de démarrage du système se trouvent dans `/boot` (ici une Centos7) :

```
# ls -lah /boot
total 72M
dr-xr-xr-x. 4 root root 4.0K Aug 28 15:53 .
dr-xr-xr-x. 17 root root 4.0K Aug 26 11:00 ..
-rw-r--r--. 1 root root 124K Nov 19 2015 config-3.10.0-327.el7.x86_64
drwxr-xr-x. 2 root root 26 Aug 26 10:56 grub
drwx------. 6 root root 104 Aug 26 10:59 grub2
-rw-r--r--. 1 root root 42M Aug 26 10:58 initramfs-0-rescue-9504b93066e14193b0bd32e69e26e75\
d.img
-rw-------. 1 root root 17M Aug 26 11:01 initramfs-3.10.0-327.el7.x86_64kdump.img
-rw-r--r--. 1 root root 589K Aug 26 10:57 initrd-plymouth.img
-rw-r--r--. 1 root root 247K Nov 19 2015 symvers-3.10.0-327.el7.x86_64.gz
-rw-------. 1 root root 2.9M Nov 19 2015 System.map-3.10.0-327.el7.x86_64
-rwxr-xr-x. 1 root root 5.0M Aug 26 10:58 vmlinuz-0-rescue-9504b93066e14193b0bd32e69e26e75d
-rwxr-xr-x. 1 root root 5.0M Nov 19 2015 vmlinuz-3.10.0-327.el7.x86_64
-rw-r--r--. 1 root root 166 Nov 19 2015 .vmlinuz-3.10.0-327.el7.x86_64.hmac
```

- Fichier `/boot/vmlinuz-*` est le noyau Linux compressé qui sera utilisé après démarrage :

```
# ls -lh /boot/vmlinuz-*
-rwxr-xr-x. 1 root root 5.0M Aug 26 10:58 vmlinuz-0-rescue-9504b93066e14193b0bd32e69e26e75d
-rwxr-xr-x. 1 root root 5.0M Nov 19 2015 vmlinuz-3.10.0-327.el7.x86_64
```

- Fichier initrd (INITial RamDisk) est une image d'un système d'exploitation minimal initialisé au démarrage du système.

```
# mkdir /tmp/initrd
# cd /tmp/initrd/
# cp /boot/initramfs-3.10.0-327.el7.x86_64.img /tmp/initrd/initramfs-3.10.0-327.el7.x86_64.\
gz
# gunzip initramfs-3.10.0-327.el7.x86_64.gz
# cpio -id < initramfs-3.10.0-327.el7.x86_64
```

On peut vérifier les fichiers :

```
ls -l /tmp/initrd/
total 43016
lrwxrwxrwx. 1 root root 7 Aug 28 15:54 bin -> usr/bin
drwxr-xr-x. 2 root root 42 Aug 28 15:54 dev
drwxr-xr-x. 12 root root 4096 Aug 28 15:54 etc
lrwxrwxrwx. 1 root root 23 Aug 28 15:54 init -> usr/lib/systemd/systemd
-rw-------. 1 root root 44039680 Aug 26 10:59 initramfs-3.10.0-327.el7.x86_64
lrwxrwxrwx. 1 root root 7 Aug 28 15:54 lib -> usr/lib
lrwxrwxrwx. 1 root root 9 Aug 28 15:54 lib64 -> usr/lib64
drwxr-xr-x. 2 root root 6 Aug 28 15:54 proc
drwxr-xr-x. 2 root root 6 Aug 28 15:54 root
drwxr-xr-x. 2 root root 6 Aug 28 15:54 run
lrwxrwxrwx. 1 root root 8 Aug 28 15:54 sbin -> usr/sbin
-rwxr-xr-x. 1 root root 3041 Aug 28 15:54 shutdown
```

```
drwxr-xr-x. 2 root root 6 Aug 28 15:54 sys
drwxr-xr-x. 2 root root 6 Aug 28 15:54 sysroot
drwxr-xr-x. 2 root root 6 Aug 28 15:54 tmp
drwxr-xr-x. 7 root root 61 Aug 28 15:54 usr
drwxr-xr-x. 2 root root 27 Aug 28 15:54 var
```

- On trouvera aussi le fichier `System.map` qui contient une table avec les symboles et leur adresse mémoire. Un symbole peut être le nom d'une variable ou d'une fonction. Cette table peut être utile pour le "crash" du noyau.

```
# head /boot/System.map-3.10.0-327.el7.x86_64
0000000000000000 A VDSO32_PRELINK
0000000000000000 D __per_cpu_start
0000000000000000 D irq_stack_union
0000000000000000 A xen_irq_disable_direct_reloc
0000000000000000 A xen_save_fl_direct_reloc
0000000000000040 A VDSO32_vsyscall_eh_frame_size
00000000000001e9 A kexec_control_code_size
00000000000001f0 A VDSO32_NOTE_MASK
0000000000000400 A VDSO32_sigreturn
0000000000000410 A VDSO32_rt_sigreturn
```

- Un fichier de configuration de compilation du noyau actuel est aussi présent dans le répertoire `/boot`.

```
# head /boot/config-3.10.0-327.el7.x86_64
#
# Automatically generated file; DO NOT EDIT.
# Linux/x86_64 3.10.0-327.el7.x86_64 Kernel Configuration
#
CONFIG_64BIT=y
CONFIG_X86_64=y
CONFIG_X86=y
CONFIG_INSTRUCTION_DECODER=y
CONFIG_OUTPUT_FORMAT="elf64-x86-64"
CONFIG_ARCH_DEFCONFIG="arch/x86/configs/x86_64_defconfig"
```

Deuxième partie Installation de logiciels

Objectifs de certification

RHCSA EX200

- **6. Déployer, configurer et gérer des systèmes**
  - 6.5. Installer et mettre à jour des paquetages logiciels depuis Red Hat Network, un référentiel distant, ou depuis le système de fichiers local
  - 6.7. Travailler avec le gestionnaire de paquets courant

LPIC 1

- *Sujet 102 : Installation de Linux et gestion de paquetages*
  - 102.3 Gestion des bibliothèques partagées
  - 102.4 Utilisation du gestionnaire de paquetage Debian
  - 102.5 Utilisation des gestionnaires de paquetage RPM et YUM

LPIC 2

- *Sujet 201 : le noyau Linux*
  - 201.2 Compilation du noyau (valeur : 3)
- *Sujet 206 : Maintenance système*
  - 206.1 Compilation et installation de programmes à partir des sources (valeur : 2)

Introduction

Cette partie aborde le vaste sujet de la gestion des logiciels sous Linux. Un premier chapitre explique le principe des installateurs principaux des distributions basées Red Hat et Debian sans éluder les autres systèmes. Un second chapitre s'attache au processus d'une compilation par les sources de logiciels sous Linux. Enfin, le propos se termine sur la mise en place de dépôt de paquetages et les installations automatiques.

Références

- https ://fr.wikipedia.org/wiki/Advanced_Packaging_Tool
- https ://help.ubuntu.com/community/SwitchingToUbuntu/FromLinux/RedHatEnterpriseLinuxAndFedora
- https ://traduc.org/LPI/Suivi/LPI101/Document/Installation_logiciels
- https ://www.kernel.org/category/signatures.html
- https ://www.cyberciti.biz/faq/debian-ubuntu-building-installing-a-custom-linux-kernel/
- https ://debian-handbook.info/browse/fr-FR/stable/sect.kernel-compilation.html
- https ://www.certdepot.net/rhel7-set-local-repository-lab/
- https ://lists.samba.org/archive/samba/2016-July/201073.html
- https ://wikigentoo.ksiezyc.pl/TIP_Converting_from_or_to_Debian.htm#Arch_Linux_7
- https ://wiki.alpinelinux.org/wiki/Comparison_with_other_distros#Update_a_particular_package
- https ://www.microhowto.info/howto/perform_an_unattended_installation_of_a_debian_package.html
- https ://doc.ubuntu-fr.org/migration
- https ://doc.ubuntu-fr.org/tutoriel/creer_un_miroir_de_depot
- https ://doc.ubuntu-fr.org/apt-cacher et https ://help.ubuntu.com/community/Apt-Cacher-Server

2. Paquets Linux

Ce chapitre est consacré à la gestion des paquets de logiciels sous Linux. On y verra la manipulation courante des utilitaires de base comme dpkg ou rpm mais aussi des gestionnaires de dépôt comme apt ou yum. On élargira cette vue avec d'autres outils comme opkg, pacman, emerge ou encore apk.

1. Gestionnaire de paquets

1.1. Gestionnaire de paquets

Un gestionnaire de paquets est un (ou plusieurs) outil(s) automatisant le processus d'installation, désinstallation, mise à jour de logiciels installés sur un système informatique.

Un paquet est une archive comprenant les fichiers informatiques, les informations et procédures nécessaires à l'installation d'un logiciel sur un système d'exploitation, en s'assurant de la cohérence fonctionnelle du système ainsi modifié.

1.2. Utilité

Le gestionnaire de paquets permet d'effectuer différentes opérations sur les paquets disponibles :

- Installation, mise à jour, et désinstallation ;
- Utilisation des paquets provenant de supports variés (CD d'installation, dépôts sur internet, partage réseau ...) ;
- Vérification des sommes de contrôle de chaque paquet récupéré pour en vérifier l'intégrité ;
- Vérification des dépendances logicielles afin d'obtenir une version fonctionnelle d'un paquetage

1.3. Nomenclature des systèmes de paquets

On trouve deux grands types de système de paquets selon les grandes familles de distributions Linux :

- RPM : Redhat Enterprise Linux, Fedora, Centos, ...
- DPKG : Debian, Ubuntu, Mint, Raspbian, ...

D'autres systèmes méritent l'intérêt :

- Portage/emerge : Gentoo
- Pacman : Archlinux
- opkg : OpenWRT

1.4. Utilitaire dpkg

`Dpkg` est utilisé pour installer, supprimer et fournir des informations à propos des paquets `*.deb` qui sont supportés par les distributions basées Debian.

Outil de bas niveau, `dpkg -i` / `dpkg -r` permettent d'installer ou de désinstaller des fichiers .deb. Pour ces tâches, on préfère utiliser des outils plus avancés comme `aptitude` ou `apt-get`, `apt-cache`.

Commandes utiles dpkg

Pour lister tous les paquets installés avec des droits priviliégés :

```
dpkg -l
```

ou

```
dpkg --get-selections
```

Pour vérifier qu'un paquet soit installé :

```
dpkg -s wget
```

Pour lister les fichiers installés par un paquet :

```
dpkg -L wget
```

Pour reconfigurer un paquet installé :

```
dpkg-reconfigure locales
```

Plus de détails sur dpkg

Le Manuel de l'administrateur debian, chapitre 5 "Système de paquetage, outils et principes fondamentaux" offre des détails et des exemples à titre d'exercice sur le sujet :

1.5. Utilitaire rpm

RPM est l'autre système de base. Il permet d'installer, mettre à jour, désinstaller, vérifier et rechercher des paquets, avec les droits de l'utilisateur `root`.

Pour Installer un paquet :

```
rpm -ivh fichier.rpm
```

Pour mettre à jour un paquet

```
rpm -Uvh fichier.rpm
```

Pour désinstaller un paquet :

```
rpm -evv fichier.rpm
```

Vérifier la signature d'un paquet :

```
rpm --checksig fichier.rpm
```

Commande rpm -q

Lister tous les paquets installés :

```
rpm -qa
```

Vérifier qu'un paquet est installé :

```
rpm -q wget
```

Lister les fichiers d'un paquet installé :

```
rpm -ql wget
```

Obtenir toutes les informations concernant un paquet installé :

```
rpm -qi wget
```

Obtenir toutes les informations concernant un paquet avant le l'installer :

```
rpm -qip fichier.rpm
```

2. Dépôt de paquets

Un gestionnaire de paquet avancé comme `apt` ou `yum` gère des sources de logiciels (la plupart du temps déjà compilés) et leur authenticité.

Le lieu où sont placés ses sources est appelé dépôt de paquet. Cette source est la plupart du temps une source locale comme un CD ou un DVD, un serveur Internet HTTP/FTP ou encore un miroir de dépôt local.

La définition d'un dépôt de paquets dépend outre de la source elle-même de la distribution, de l'architecture matérielle, des sources officielles ou autres.

Certains concepteurs de logiciels fabriquement eux-mêmes les binaires d'installation pour les distributions et hébergent leurs propres dépôts de paquets.

2.1. Principe de fonctionnement

Principe de fonctionnement d'un gestionnaire de paquet avancé :

- Les logiciels disponibles sont contenus dans une liste qui doit être à jour afin d'assurer la cohérence de l'ensemble du système.
- Au moment de la demande d'installation, cette liste est consultée pour prendre les fichiers nécessaires.
- Le système de paquetage décompresse et place les différents fichiers binaires, de configuration et de documentation aux endroits appropriés. Éventuellement, un dialogue de configuration intervient.
- *Éventuellement*, le système de paquetage installe automatiquement un service et le démarre.

Tâches

- Vérification de l'existence d'un paquet
- Version du logiciel dans le paquet
- Fichiers de configuration
- Source
- Fichiers de configuration /etc
- Désinstallation
- Purge des fichiers
- Suppression des dépendances orphelines

2.2. APT

APT simplifie l'installation, la mise à jour et la désinstallation de logiciels en automatisant la récupération de paquets à partir de sources APT (sur Internet, le réseau local, des CD-ROM, etc.), la gestion des dépendances et parfois la compilation.

Lorsque des paquets sont installés, mis à jour ou enlevés, les programmes de gestion de paquets peuvent afficher les dépendances des paquets, demander à l'administrateur si des paquets recommandés ou suggérés par des paquets nouvellement installés devraient aussi être installés, et résoudre les dépendances automatiquement. Les programmes de gestion de paquets peuvent aussi mettre à jour tous les paquets.

APT est essentiellement une bibliothèque C++ de fonctions utilisées par plusieurs programmes de gestion de paquets. Un de ces programmes est `apt`, probablement le plus connu et celui recommandé officiellement par le projet Debian. aptitude est également populaire et propose des fonctionnalités étendues par rapport à `apt-get`. Aujourd'hui, `apt` est une commande qui se suffit à elle-même pour la plupart des opérations concernant les paquets Debian/Ubuntu.

Sources APT

Les sources à partir desquelles `apt` va chercher les paquets sont définies dans le fichier `/etc/apt/sources.list`

Par exemple sur une machine Debian 7 Wheezy :

```
cat /etc/apt/sources.list
```

```
deb https://http.debian.net/debian wheezy main
deb https://http.debian.net/debian wheezy-updates main
deb https://security.debian.org wheezy/updates main
```

La section **main** comprend l'ensemble des paquets qui se conforment aux DFSG - Directives Debian pour le logiciel libre et qui n'ont pas besoin de programmes en dehors de ce périmètre pour fonctionner. Ce sont les seuls paquets considérés comme faisant partie de la distribution Debian.

La section **contrib** comprend l'ensemble des paquets qui se conforment aux DFSG, mais qui ont des dépendances en dehors de main (qui peuvent être empaquetées pour Debian dans non-free).

La section **non-free** contient des logiciels qui ne se conforment pas aux DFSG.

Par exemple sur une machine Ubuntu 16.04 Xenial :

```
cat /etc/apt/sources.list
```

```
deb https://archive.ubuntu.com/ubuntu xenial main universe
deb https://archive.ubuntu.com/ubuntu xenial-updates main universe
deb https://archive.ubuntu.com/ubuntu xenial-security main universe
```

Ubuntu maintient officiellement les paquets **main** (logiciels libres) et **restricted** (logiciels non-libres).

La communauté Ubuntu fournit les paquets **universe** (libres) et **multiverse** (non-libres).

On prendra l'habitude de mettre à jour la liste de paquetages avec :

```
apt-get update
```

Recherche APT

Recherche dans les descriptions de paquets :

```
apt-cache search wget
```

Mais `apt search` fonctionne aussi :

```
apt search wget
```

Voir les informations d'un paquet :

```
apt-cache show wget
```

Vérifier les dépendances d'un paquet :

```
apt-cache showpkg wget
```

Mise à jour et installation avec APT

Mettre à jour tous les paquets sans ajout de nouveaux paquets :

```
apt-get update && apt-get upgrade
```

ou :

```
apt update && apt upgrade
```

Mettre à jour tous les paquets installés vers les dernières versions en installant de nouveaux paquets si nécessaire :

```
apt dist-upgrade
```

Installation ou mise-à-jour d'un paquet :

```
apt install wget
```

Installation sans dialogue :

```
apt -y install wget
```

Désinstallation de paquets APT

Retirer le paquets sans les configurations :

```
apt remove wget
```

Retirer le paquets sans les dépendances :

```
apt autoremove wget
```

Retirer totalement un paquet :

```
apt purge wget
```

On peut combiner les deux :

```
apt autoremove --purge wget
```

Retire les dépendances non nécessaires :

```
apt autoremove
```

Suppression des fichiers mis en cache dans `var/cache/apt/archives` :

```
apt clean
```

Utilement, on ira lire les précisions la section 6.2, la section 6.3 et la section 6.4 du Manuel de l'Administrateur Debian :

Réinstaller un paquet :

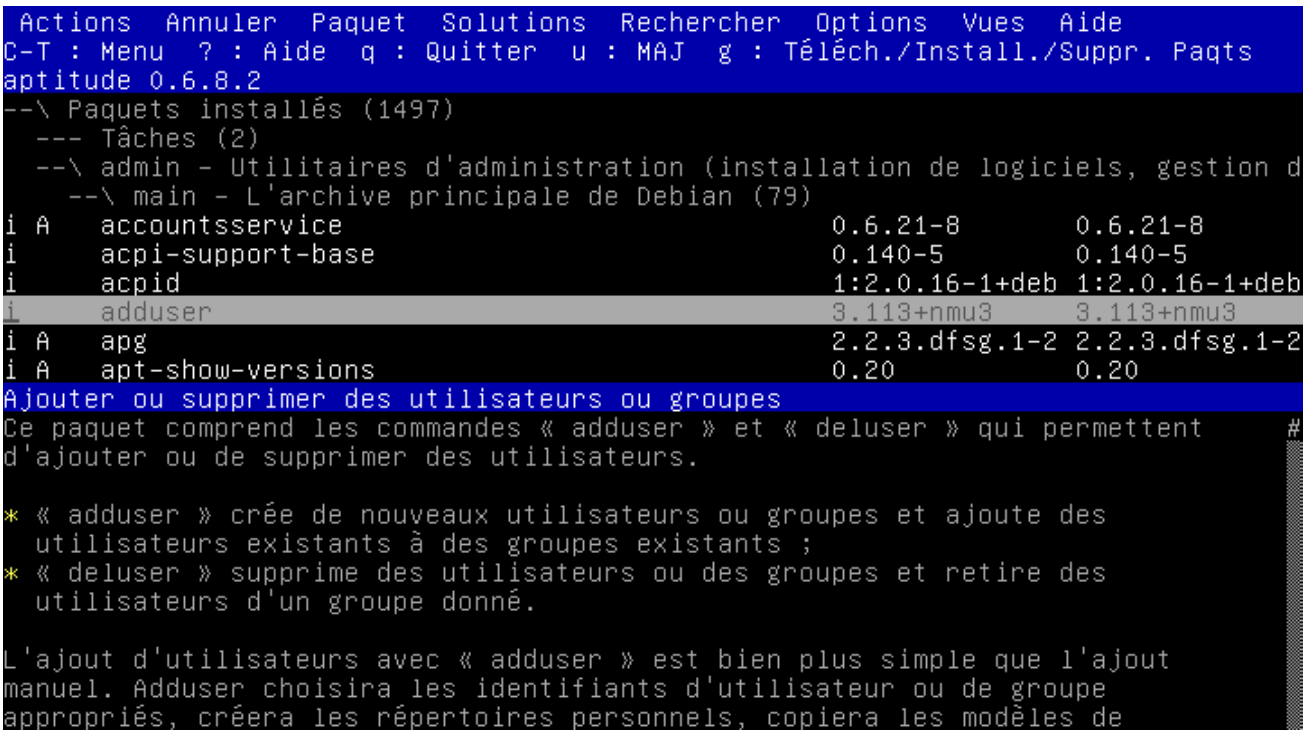
```
apt --reinstall install postfix
```

Installation d'une version "unstable" :

```
apt install spamassassin/unstable
```

`apt full-upgrade` est également la commande employée par ceux qui exploitent quotidiennement la version Unstable de Debian et suivent ses évolutions au jour le jour. Elle est si simple qu'elle parle d'elle-même : c'est bien cette fonctionnalité qui a fait la renommée d'APT.

`aptitude` est un programme interactif en mode semi-graphique, utilisable sur la console, qui permet de naviguer dans la liste des paquets installés et disponibles, de consulter l'ensemble des informations et de les marquer en vue d'une installation ou d'une suppression. Comme il s'agit cette fois d'un programme réellement conçu pour être utilisé par les administrateurs, on y trouve des comportements par défaut plus intelligents que dans apt-get, en plus d'une interface plus abordable.

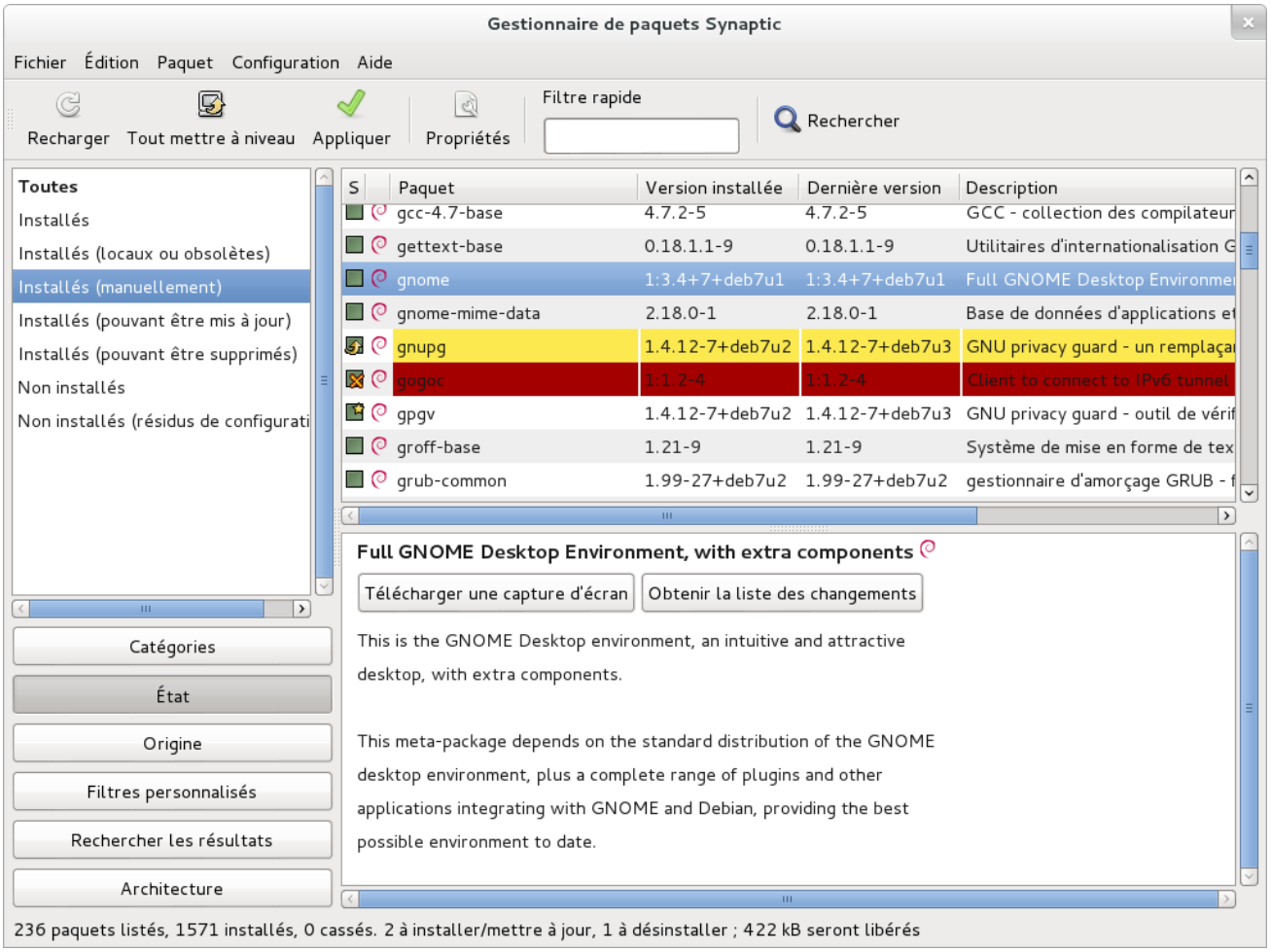


Gestionnaire de paquets aptitude

Source de l'image

Gestionnaire de paquets aptitude

synaptic est un gestionnaire de paquets Debian en mode graphique (il utilise GTK+/GNOME). Il dispose d'une interface graphique efficace et propre. Ses nombreux filtres prêts à l'emploi permettent de voir rapidement les nouveaux paquets disponibles, les paquets installés, ceux que l'on peut mettre à jour, les paquets obsolètes, etc. En naviguant ainsi dans les différentes listes, on indique progressivement les opérations à effectuer (installer, mettre à jour, supprimer, purger). Un simple clic suffit à valider l'ensemble de ces choix et toutes les opérations enregistrées sont alors effectuées en une seule passe.



Gestionnaire de paquets synaptic

Source de l'image

Gestionnaire de paquets synaptic

Authentification des paquets Debian

Debian offre un moyen de s'assurer que le paquet installé provient bien de son mainteneur et qu'il n'a subi aucune modification par un tiers : il existe un mécanisme de scellement des paquets.

Cette signature n'est pas directe : le fichier signé est un fichier `Release` placé sur les miroirs Debian et qui donne la liste des différents fichiers `Packages` (y compris sous leurs formes compressées `Packages.gz` et `Packages.xz` et les versions incrémentales), accompagnés de leurs sommes de contrôle MD5, SHA1 et SHA256 (pour vérifier que leur contenu n'a pas été altéré). Ces fichiers Packages renferment à leur tour une liste de paquets Debian et leurs sommes de contrôle, afin de garantir que leur contenu n'a pas lui non plus été altéré.*

La gestion des clés de confiance se fait grâce au programme `apt-key`, fourni par le paquet `apt`. Ce programme maintient à jour un trousseau de clés publiques GnuPG, qui sont utilisées pour vérifier les signatures des fichiers `Release.gpg` obtenus depuis les miroirs Debian.

Il est possible de l'utiliser pour ajouter manuellement des clés supplémentaires (si l'on souhaite ajouter des miroirs autres que les miroirs officiels) ; mais dans le cas le plus courant, on n'a besoin que des clés officielles Debian, qui sont automatiquement maintenues à jour par le paquet `debian-archive-keyring` (qui installe les trousseaux de clés dans '/etc/apt/trusted.gpg.d).

Cependant, la première installation de ce paquet est également sujette à caution, car même s'il est signé comme les autres paquets, cette signature ne peut pas être vérifiée extérieurement. On s'attachera donc à vérifier les

empreintes (fingerprints) des clés importées, avant de leur faire confiance pour installer de nouveaux paquets avec `apt-key fingerprint`.

Source : Vérification d'authenticité des paquets

Par exemple,

```
apt-key fingerprint
```

```
/etc/apt/trusted.gpg.d/debian-archive-jessie-automatic.gpg
-----------------------------------------------------------
pub   4096R/2B90D010 2014-11-21 [expire : 2022-11-19]
 Empreinte de la clef = 126C 0D24 BD8A 2942 CC7D  F8AC 7638 D044 2B90 D010
uid                  Debian Archive Automatic Signing Key (8/jessie) <ftpmaster@debian.org>

/etc/apt/trusted.gpg.d/debian-archive-jessie-security-automatic.gpg
-------------------------------------------------------------------
pub   4096R/C857C906 2014-11-21 [expire : 2022-11-19]
 Empreinte de la clef = D211 6914 1CEC D440 F2EB  8DDA 9D6D 8F6B C857 C906
uid                  Debian Security Archive Automatic Signing Key (8/jessie) <ftpmaster@de\
bian.org>

/etc/apt/trusted.gpg.d/debian-archive-jessie-stable.gpg
-------------------------------------------------------
pub   4096R/518E17E1 2013-08-17 [expire : 2021-08-15]
 Empreinte de la clef = 75DD C3C4 A499 F1A1 8CB5  F3C8 CBF8 D6FD 518E 17E1
uid                  Jessie Stable Release Key <debian-release@lists.debian.org>

/etc/apt/trusted.gpg.d/debian-archive-squeeze-automatic.gpg
-----------------------------------------------------------
pub   4096R/473041FA 2010-08-27 [expire : 2018-03-05]
 Empreinte de la clef = 9FED 2BCB DCD2 9CDF 7626  78CB AED4 B06F 4730 41FA
uid                  Debian Archive Automatic Signing Key (6.0/squeeze) <ftpmaster@debian.o\
rg>

/etc/apt/trusted.gpg.d/debian-archive-squeeze-stable.gpg
--------------------------------------------------------
pub   4096R/B98321F9 2010-08-07 [expire : 2017-08-05]
 Empreinte de la clef = 0E4E DE2C 7F3E 1FC0 D033  800E 6448 1591 B983 21F9
uid                  Squeeze Stable Release Key <debian-release@lists.debian.org>

/etc/apt/trusted.gpg.d/debian-archive-wheezy-automatic.gpg
----------------------------------------------------------
pub   4096R/46925553 2012-04-27 [expire : 2020-04-25]
 Empreinte de la clef = A1BD 8E9D 78F7 FE5C 3E65  D8AF 8B48 AD62 4692 5553
uid                  Debian Archive Automatic Signing Key (7.0/wheezy) <ftpmaster@debian.or\
g>

/etc/apt/trusted.gpg.d/debian-archive-wheezy-stable.gpg
-------------------------------------------------------
pub   4096R/65FFB764 2012-05-08 [expire : 2019-05-07]
 Empreinte de la clef = ED6D 6527 1AAC F0FF 15D1  2303 6FB2 A1C2 65FF B764
uid                  Wheezy Stable Release Key <debian-release@lists.debian.org>
```

Empêcher le démarrage d'un service après une installation

La création d'un script de sortie `/usr/sbin/policy-rc.d` empêchera le lancement du service après installation.

```
cat > /usr/sbin/policy-rc.d << EOF
#!/bin/sh
echo "All runlevel operations denied by policy" >&2
exit 101
EOF
chmod +x /usr/sbin/policy-rc.d
```

L'existence de ce script donnera ce message après une installation :

```
All runlevel operations denied by policy
invoke-rc.d: policy-rc.d denied execution of start.
```

2.3. YUM / DNF

"Yum", pour "Yellowdog Updater Modified", est un gestionnaire de paquets pour des distributions Linux telles que Fedora et Red Hat Enterprise Linux, créé par Yellow Dog Linux.

"Dandified Yum" ou "DNF" est le successeur de YUM dont l'usage est en quasiment identique depuis Red Hat Enterprise Linux 8 et Fedora 22.

YUM et DNF permettent de gérer l'installation et la mise à jour des logiciels installés sur une distribution de type Red Hat. Ce sont des surcouches de RPM gérant les téléchargements et les dépendances, de la même manière que APT de Debian.

YUM commandes de base

> Contrairement à APT, YUM met à jour sa liste de paquets automatiquement.

Chercher un paquet :

```
yum search wget
```

Lister des informations concernant un paquet :

```
yum list wget
yum info wget
```

Installer un paquet :

```
yum install wget
```

Installer un paquet sans dialogue :

```
yum -y install wget
```

Désinstaller un paquet

```
yum remove wget
```

YUM mise-à-jour

Mise-à-jour d'un paquet :

```
yum update wget
```

Vérification des mise-à-jours disponibles :

```
yum check-update
```

Mise-à-jours de sécurité et des binaires :

```
yum update
```

YUM Group Packages

Les groupes de paquets sont des collections de paquets :

```
yum groups list
yum groups info group
yum groups install group
yum groups update group
yum groups remove group
```

YUM dépôts de paquets

Liste des dépôts de paquets :

```
yum repolist
yum repolist all
```

Installer un dépôt supplémentaire EPEL (Extra Packages for Enterprise Linux) :

```
yum install epel-release
```

Installer un dépôt supplémentaire

La configuration des dépôts est située dans le dossier `/etc/yum.repos.d/` :

```
ls /etc/yum.repos.d/
```

```
CentOS-Base.repo  CentOS-Debuginfo.repo  CentOS-Media.repo    CentOS-Vault.repo
CentOS-CR.repo    CentOS-fasttrack.repo  CentOS-Sources.repo
```

Par exemple le premier dépôt configuré dans le fichier `CentOS-Base.repo` :

```
[base]
name=CentOS-$releasever - Base
mirrorlist=https://mirrorlist.centos.org/?release=$releasever&arch=$basearch&repo=os&infra=\
$infra
#baseurl=https://mirror.centos.org/centos/$releasever/os/$basearch/
gpgcheck=1
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-CentOS-7
```

On notera une déclaration de section `[base]` en en-tête et quatre variables essentielles :

- `name` qui indique le nom du dépôt.
- `mirrorlist` ou `baseurl` qui indiquent l'emplacement du dépôt
- `gpgckeck` qui demande une vérification d'intégrité et `gpgkey` qui fixe le fichier de vérification d'emprunte. Il n'est pas nécessaire `gpckeck=0` est configuré.
- Enfin, `enabled=1`activerait la prise en compte du dépôt

L'utilitaire `yum-config-manager` permet d'ajouter un dépôt aisément :

```
yum-config-manager --add-repo=https://ftp.belnet.be/ftp.centos.org/7/os/x86_64/
```

```
Modules complémentaires chargés : fastestmirror, langpacks
adding repo from: https://ftp.belnet.be/ftp.centos.org/7/os/x86_64/

[ftp.belnet.be_ftp.centos.org_7_os_x86_64_]
name=added from: https://ftp.belnet.be/ftp.centos.org/7/os/x86_64/
baseurl=https://ftp.belnet.be/ftp.centos.org/7/os/x86_64/
enabled=1
```

Y ajouter `gpgcheck=0` :

```
echo "gpgcheck=0" >> /etc/yum.repos.d/ftp.belnet.be_ftp.centos.org_7_os_x86_64_.repo
cat /etc/yum.repos.d/ftp.belnet.be_ftp.centos.org_7_os_x86_64_.repo
```

```
[ftp.belnet.be_ftp.centos.org_7_os_x86_64_]
name=added from: https://ftp.belnet.be/ftp.centos.org/7/os/x86_64/
baseurl=https://ftp.belnet.be/ftp.centos.org/7/os/x86_64/
enabled=1

gpgcheck=0
```

Remettre à jour la liste des paquetages :

```
yum clean all
yum repolist
```

YUM gestion des paquets

Lister les paquets installés :

```
yum list installed | less
```

Effacer le cache /var/cache/yum/ :

```
yum clean all
```

Historique des transactions yum :

```
yum history
```

2.4. Autres logiciel de gestion des paquets

- Pacman : Arch Linux.
- Emerge : Gentoo
- Opkg : Openwrt

3. Maintenance et mises à jour

3.1. Maintenance des mises à jour d'un système Debian

`apticron`, dans le paquet du même nom. Il s'agit simplement d'un script, appelé quotidiennement par `cron`, qui met à jour la liste des paquets disponibles et envoie un courrier électronique à une adresse donnée pour lister les paquets qui ne sont pas installés dans leur dernière version, ainsi qu'une description des changements qui ont eu lieu. Ce script vise principalement les utilisateurs de Debian Stable, on s'en doute.

On pourra donc tirer parti du script `/etc/cron.daily/apt`, installé par le paquet apt. Ce script est lui aussi lancé quotidiennement par cron, donc sans interface interactive. Pour contrôler son fonctionnement, on utilisera des variables de configuration d'APT (qui seront donc stockées dans un fichier sous `/etc/apt/apt.conf.d/`). Les plus importantes sont :

- `APT ::Periodic ::Update-Package-Lists`
- `APT ::Periodic ::Download-Upgradeable-Packages`
- `APT ::Periodic ::AutocleanInterval`
- `APT ::Periodic ::Unattended-Upgrade`

```
APT::Periodic::Update-Package-Lists
```

Cette option permet de spécifier une fréquence (en jours) de mise à jour des listes de paquets. Si l'on utilise `apticron`, on pourra s'en passer, puisque cela ferait double emploi.

`APT::Periodic::Download-Upgradeable-Packages`

Cette option spécifie également une fréquence en jours, qui porte sur le téléchargement des paquets mis à jour. Là encore, les utilisateurs d'apticron pourront s'en passer.

`APT::Periodic::AutocleanInterval`

Cette option couvre une fonction que n'a pas `apticron` : elle spécifie la fréquence à laquelle le cache d'APT pourra être automatiquement épuré des paquets obsolètes (ceux qui ne sont plus disponibles sur les miroirs ni référencés par aucune distribution). Elle permet de ne pas avoir à se soucier de la taille du cache d'APT, qui sera ainsi régulée automatiquement.

`APT::Periodic::Unattended-Upgrade`

Lorsque cette option est activée, le script quotidien exécutera unattended-upgrade (dans le paquet `unattended-upgrades`) qui, comme son nom l'indique, automatise le processus de mise à jour pour certains paquets ; par défaut, il ne s'occupe que des mises à jour de sécurité, mais cela est configurable dans le fichier `/etc/apt/apt.conf.d/50unattended-upgrades`). Notons que cette option peut être activée avec `debconf`, à l'aide de la commande `dpkg-reconfigure -plow unattended-upgrades`.

Source : Maintenir un système à jour

3.2. Mise à jour d'une distribution Debian depuis une ancienne version

Mise à jour depuis Debian 7 (wheezy) vers Debian 8 (jessie)

Recommendations :

- Effacer les paquets non nécessaires
- Mettre à jour le système actuel
- Réaliser une sauvegarde des données

Mettre à jour la distribution Debian Wheezy :

```
apt-get update
apt-get upgrade
apt-get dist-upgrade
```

Mettre à jour les sources d'installation :

```
sed -i 's/wheezy/jessie/g' /etc/apt/sources.list
```

Mettre à jour les paquets :

```
apt-get update
apt-get -y upgrade
```

Mettre à jour la distribution :

```
apt-get -y dist-upgrade
```

Redémarrer :

```
reboot
```

Vérifier la version :

```
hostnamectl
```

```
   Static hostname: wheezy1
         Icon name: computer-vm
           Chassis: vm
        Machine ID: cab21b38a8058c4d3f6641f1587fa5b7
           Boot ID: cde0bd1e7ada4c44acd12bae10adff75
    Virtualization: kvm
  Operating System: Debian GNU/Linux 8 (jessie)
            Kernel: Linux 3.16.0-4-amd64
      Architecture: x86-64
```

Remettre à jour :

```
apt-get update
apt-get -y upgrade
apt-get -y autoremove
apt-get -y dist-upgrade
```

Mise à jour depuis Debian 8 (jessie) vers Debian 9 (stretch)

La procédure est identique de la version Debian 8 Jessie à la version Debian 9 Stretch.

```
cp /etc/apt/sources.list /etc/apt/sources.list_backup
sed -i 's/jessie/stretch/g' /etc/apt/sources.list
apt-get update
apt-get -y upgrade
apt-get -y dist-upgrade
reboot
```

```
apt-get update
apt-get -y upgrade
apt-get -y autoremove
apt-get -y dist-upgrade
```

```
hostnamectl
```

```
   Static hostname: wheezy1
         Icon name: computer-vm
           Chassis: vm
        Machine ID: cab21b38a8058c4d3f6641f1587fa5b7
           Boot ID: efd16e22f98a42d0a7d3cf44dba21fc9
    Virtualization: kvm
  Operating System: Debian GNU/Linux 9 (stretch)
            Kernel: Linux 4.8.0-2-amd64
      Architecture: x86-64
```

Mise-à-jour de versions Ubuntu

Source : https ://doc.ubuntu-fr.org/migration

Fortement déconseillée, la procédure propose de passer de révision en révision. Une sauvegarde du système est alors recommandée.

L'outil en ligne de commande `do-release-upgrade` permet d'effectuer une mise à niveau d'Ubuntu sans utiliser d'utilitaire graphique. Il est particulièrement pertinent pour les serveurs, qui fonctionnent sans interface graphique. L'ensemble des options de cet outil peut être lue en exécutant la commande :

```
do-release-upgrade --help
```

Voici quelques-unes des options les plus utiles :

```
do-release-upgrade --check-dist-upgrade-only
```

L'option `–check-dist-upgrade-only` vérifie l'existence d'une nouvelle version. Si une nouvelle version est trouvée, celle-ci est affichée en résultat dans le terminal. Exécutée ainsi, cette commande n'effectue qu'une vérification ; aucune mise à niveau n'est faite.

```
do-release-upgrade --sandbox
```

L'option `–sandbox` permet de tester une mise à niveau dans un environnement protégé. Ceci est particulièrement utile pour tester le déploiement d'une mise à niveau avant de procéder à son application dans l'environnement de production.

```
do-release-upgrade
```

Sans option, l'outil do-release-upgrade recherche et procède à une mise à niveau vers la prochaine version LTS ou stable disponible, si elle existe.

4. Comparatif des gestionnaires de paquets par distribution

Du point de vue de l'administrateur système, les distributions Linux *peuvent* se distinguer par :

1. le gestionnaire et le sytème de paquets
2. les scripts d'initialisation et les niveaux d'exécution
3. le chargeur de démarrage
4. l'emplacement des fichiers de configuration du réseau et des dépôts

On s'intéressa ici aux différences "génétiques" concernant la gestion des paquets.

4.1. Debian/Ubuntu c. Fedora/RHEL/SL/Centos

Action	Debian/Ubuntu	Fedora/RHEL/SL/Centos
1. Mise à jour de la liste des paquets	`apt-get update`	`yum update`, `yum check-update`
2. Affichage des mises-à-jour disponibles	`apt-get upgrade --simulate`	`yum list updates`
3. Installation de paquets spécifiques	`apt-get install package1 package2`	`yum install package1 package2`
4. Réinstallation d'un paquet	`apt-get install --reinstall package`	`yum reinstall package`
5. Mise à jour d'un paquet	`apt-get upgrade package1 package2`	`yum update package`
6. Mise à jour du système	`apt-get upgrade`, `apt-get dist-upgrade`, `apt upgrade`, `apt full-upgrade`	`yum upgrade`
7. Recherche de paquets	`apt-cache search searchword`, `apt-cache search --full --names-only searchword`	`yum search searchword`
8. Liste de paquets installés	`dpkg -l`, `apt list --installed`	`rpm -qa`
9. Information sur un paquet	`apt-cache show package`, `apt show package`, `dpkg -s package`	`yum info package`, `yum list package`, `yum deplist package`
10. Désinstaller des paquets	`apt-get remove --purge package1 package2`, `apt-get autoremove`	`yum remove package1 package2`
11. Téléchargement de paquets sans installation	`apt-get install --download-only package1 package2`	`yum install --downloadonly --downloaddir=<directory> <package>`
12. Effacement des paquets téléchargés	`apt-get clean`, `apt-get clean` (paquets dépassés)	`yum clean all`
13. Configuration des dépôts	`/etc/apt/sources.list`	`/etc/yum.repos.d/`

4.2. Alpine Linux c. Arch Linux c. Gentoo

Action	Alpine Linux	Arch Linux	Gentoo
1. Mise à jour de la liste des paquets	`apk update`	`pacman -Sy`	`emerge --sync`
2. Affichage des mises-à-jour disponibles	`apk version -v` ou `apk version -v -l '<'`	`pacman -Qu`	`emerge -Dupv world` ou `emerge --deep --update --pretend world`
3. Installation de paquets spécifiques	`apk add package1 package2`	`pacman -S package1 package2`	`emerge package1 package2`
4. Réinstallation d'un paquet	`apk del package1 && apk add package1`	`pacman -Sf package1 package2`	`emerge --oneshot package1`
5. Mise à jour d'un paquet	`apk add -u package1 package2`	`pacman -S package1 package2`	`emerge --update package1 package2`
6. Mise à jour du système	-	`pacman -Syu`	-

Action	Alpine Linux	Arch Linux	Gentoo
7. Recherche de paquets	`apk search searchword`	`pacman -Ss searchword`, `pacman -Si packagename`	`emerge --searchdesc searchword`, `eix searchword`, `esearch searchword` `yum search searchword`
8. Liste de paquets installés	`apk info`	`pacman -Qs`, `pacman -Q`, `pacman -Q`	`emerge gentoolkit && equery list`
9. Information sur un paquet	`apk info -a package`	`pacman -Si package`	-
10. Désinstaller des paquets	`apk del package1 package2`	`pacman -R package1 package2`	`emerge --depclean package1 package2`
11. Téléchargement de paquets sans installation	`apk fetch package1 package2`	`pacman -Sw package1 package2`	`emerge --fetchonly package1 package2`
12. Effacement des paquets téléchargés	*Automatique*	*Automatique*	`rm -rf /usr/portage/distfiles/*`
13. Configuration des dépôts	`/etc/apk/repositories`	`/etc/opkg.conf`	`etc/portage/repos.conf/gentoo.conf`, et bien plus

4.3. OpenWRT

Action	OpenWRT
1. Mise à jour de la liste des paquets	`opkg update`
2. Affichage des mises-à-jour disponibles	`opkg list-upgradable`
3. Installation de paquets spécifiques	`opkg install <pkgs ou FQDN>`
4. Réinstallation d'un paquet	`opkg install --force-reinstall <pkgs>`
5. Mise à jour d'un paquet	`opkg upgrade <pkgs>` (non recommandé)
6. Mise à jour du système	-
7. Recherche de paquets	`opkg list [pkg ou globp]`, `opkg search <file ou globp>`
8. Liste de paquets installés	`opkg list-installed`
9. Information sur un paquet	`opkg info [pkg ou globp]`, `opkg status [pkg ou globp]`
10. Désinstaller des paquets	`opkg remove <pkgs ou globp>`
11. Téléchargement de paquets sans installation	`opkg --download-only download <pkg>`
12. Effacement des paquets téléchargés	option `--force-removal-of-dependent-packages`
13. Configuration des dépôts	`/etc/opkg.conf`

5. Mettre à jour le noyau

5.1. Procédure RHEL

```
yum update kernel
```

ou alors si le fichier rpm est disponible

```
rpm -ivh kernel.rpm
```

Le dernier noyau installé devient le premier par défaut :

```
grub2-editenv list
```

```
saved_entry=CentOS Linux (3.10.0-327.13.1.el7.x86_64) 7 (Core)
```

```
grep ^menuentry /boot/grub2/grub.cfg
```

```
menuentry 'CentOS Linux (3.10.0-327.13.1.el7.x86_64) 7 (Core)' --class centos --class gnu-l\
inux --class gnu --class os --unrestricted $menuentry_id_option 'gnulinux-3.10.0-327.el7.x8\
6_64-advanced-5cc65046-7a0e-450b-99e8-f0cc34954d75' {
menuentry 'CentOS Linux (3.10.0-327.el7.x86_64) 7 (Core)' --class centos --class gnu-linux \
--class gnu --class os --unrestricted $menuentry_id_option 'gnulinux-3.10.0-327.el7.x86_64-\
advanced-5cc65046-7a0e-450b-99e8-f0cc34954d75' {
menuentry 'CentOS Linux (0-rescue-d939e80ee5d6473297b10a3839c85928) 7 (Core)' --class cento\
s --class gnu-linux --class gnu --class os --unrestricted $menuentry_id_option 'gnulinux-0-\
rescue-d939e80ee5d6473297b10a3839c85928-advanced-5cc65046-7a0e-450b-99e8-f0cc34954d75' {
```

Modifier le noyau par défaut :

```
grub2-set-default 0
```

Générer la configuration :

```
grub2-mkconfig -o /boot/grub2/grub.cfg
```

```
Generating grub configuration file ...
Found linux image: /boot/vmlinuz-3.10.0-327.13.1.el7.x86_64
Found initrd image: /boot/initramfs-3.10.0-327.13.1.el7.x86_64.img
Found linux image: /boot/vmlinuz-3.10.0-327.el7.x86_64
Found initrd image: /boot/initramfs-3.10.0-327.el7.x86_64.img
Found linux image: /boot/vmlinuz-0-rescue-d939e80ee5d6473297b10a3839c85928
Found initrd image: /boot/initramfs-0-rescue-d939e80ee5d6473297b10a3839c85928.img
done
```

5.2. Procédure Ubuntu

‼–‼

Troisième partie Scripts Shell

Objectifs de certification

Linux Essentials

- *Sujet 3 : Le pouvoir de la ligne de commande*
  - 3.3 Conversion de commandes en script (valeur : 4)

LPIC 1

- *Sujet 105 : Shells et scripts Shell*
  - 105.1 Personnalisation et utilisation de l'environnement du shell
  - 105.2 Personnalisation ou écriture de scripts simples

RHCSA EX200

- **2. Créer des scripts shell simples**
  - 2.1. Exécution conditionnelle du code (utilisation de : `if`, `test`, `[]`, etc.)
  - 2.2. Utiliser des constructions en boucle (`for`, etc.) pour traiter le fichier, l'entrée en ligne de commande
  - 2.3. Traiter les entrées de script (`$1`, `$2`, etc.)
  - 2.4. Traiter les codes de sortie des commandes shell dans un script

RHCE EX300

- **1. Configuration et gestion de systèmes**
  - 1.8. Utiliser des scripts shell pour automatiser les tâches de maintenance

Introduction

Cette partie est une initiation au scripting Shell. On y aborde les notions de base des scripts Bash, les structures conditionnelles, les boucles, des concepts avancés sur les variables, ainsi que des modèles, des figures et des exemples de scripts.

3. Scripts Shell

On trouvera dans ce chapitre une initiation pratique au scripting Bash.

Les sections qui suivent dans ce chapitre s'inspirent notamment du livre Scripts shell Linux et Unix de Christophe Blaess qu'il est conseillé d'acquérir. L'ouvrage est orienté embarqué mais convient parfaitement pour un apprentissage précis, rapide, intéressant et dynamique.

1. Scripts Bash : notions

Cette section expose des rudiments pour commencer à automatiser ses tâches d'administration en Bash.

1.1. Scripts Bash

Voici une liste de départ des concepts à maîtriser pour le scripting Bash :

- shebang
- variables positionnelles
- variables internes
- fonctions et programme principal
- fin de script
- test
- conditions
- boucles
- débogage
- `~/.bashrc`
- Références et exemples

1.2. Shebang

Le shebang, représenté par `#!`, est un en-tête d'un fichier texte qui indique au système d'exploitation que ce fichier n'est pas un fichier binaire mais un script (ensemble de commandes) ; sur la même ligne est précisé l'interpréteur permettant d'exécuter ce script. Pour indiquer au système qu'il s'agit d'un script qui sera interprété par bash on placera le shebang sur la première ligne :

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

1.3. Hello World

```
#!/bin/bash
# script0.sh
echo "Hello World"
exit
```

Donner les droits d'exécution au script.

```
chmod +x script0.sh
```

1.4 Variables prépositionnées

Certaines variables ont une signification spéciale réservée. Ces variables sont très utilisées lors la création de scripts :

- pour récupérer les paramètres transmis sur la ligne de commande,
- pour savoir si une commande a échoué ou réussi,
- pour automatiser le traitement de tous paramètres.

Liste de variables prépositionnées

- `$0` : nom du script. Plus précisément, il s'agit du paramètre 0 de la ligne de commande, équivalent de `argv[0]`
- `$1`, `$2`, ..., `$9` : respectivement premier, deuxième, ..., neuvième paramètre de la ligne de commande
- `$*` : tous les paramètres vus comme un seul mot
- `$@` : tous les paramètres vus comme des mots séparés : "`$@`" équivaut à "`$1`" "`$2`" ...
- `$#` : nombre de paramètres sur la ligne de commande
- `$-` : options du shell
- `$?` : code de retour de la dernière commande
- {`$$`} : PID du shell
- `$!` : PID du dernier processus lancé en arrière-plan
- `$_` : dernier argument de la commande précédente

Par exemple :

```
#!/bin/bash
# script1.sh
echo "Nom du script $0"
echo "premier paramètre $1"
echo "second paramètre $2"
echo "PID du shell " {$$}
echo "code de retour $?"
exit
```

Donner les droits d'exécution du script par l'utilisateur :

```
chmod +ux script1.sh
```

Exécuter le script avec deux paramètres :

```
./script1.sh 10 zozo
```

1.5. Variables internes

En début de script, on peut définir la valeur de départ des variables utilisées dans le script. Elles ne sont connues que par le processus associé au lancement du script.

```
VARIABLE="valeur"
```

Elles s'appellent comme ceci dans le script :

```
echo $VARIABLE
```

Il peut être utile de marquer les limites d'une variable avec les accolades.

```
echo ${VARIABLE}
```

Par exemple :

```
#!/bin/bash
# script2.sh
PRENOM="francois"
echo "dossier personnel /home/${PRENOM}"
exit
```

1.6. Interaction utilisateur

La commande `echo` pose une question à l'utilisateur.

La commande `read` lit les valeurs entrées au clavier et les stocke dans une variable à réutiliser.

```
echo "question"
read reponse
echo $response
```

On peut aller plus vite avec `read -p` qui sort du texte et attend une valeur en entrée :

```
read -p "question" reponse
echo $reponse
```

1.7. Fonctions

Une fonction est un bloc d'instructions que l'on peut appeller ailleurs dans le script. Pour déclarer une fonction, on utilise la syntaxe suivante :

```
maFonction()
{
  echo hello world
}
```

Ou de manière plus ancienne :

```
function ma_fonction {
  echo hello world
}
```

La déclaration d'une fonction doit toujours se situer avant son appel. On mettra donc les fonctions en début de script.

Par exemple :

```
#!/bin/bash
# script3.sh
read -p "quel votre prénom ?" prenom
reponse() {
    echo $0
    echo "merci $prenom"
    exit 1
}
reponse
exit
```

2. Structures conditionnelles

2.1. Structure conditionnelle if/then

```
if condition ; then
    commande1
else
    commande2
fi
```

2.2. Tests

La condition pourra contenir un test. Deux manières de réaliser un test (avec une préférence pour la première) :

```
[ expression ]
```

ou

```
test expression
```

Note : `/user/bin/[` est renseigné comme un programme sur le système.

On peut aussi utiliser la version étendue de la commande `test` :

```
[[ expression ]]
```

Il y a beaucoup d'opérateurs disponibles pour réaliser des tests sur les fichiers, sur du texte ou sur des valeurs arithmétiques. La commande `man test` donnera une documentation à lire avec attention : tout s'y trouve.

Par exemple :

```
#!/bin/bash
# script4.sh test si $passwdir existe
passwdir=/etc/passwdd
checkdir() {
    if [ -e $passwdir ]; then
        echo "le fichier $passwdir existe"
    else
        echo "le fichier $passwdir n'existe pas"
    fi
}
checkdir
exit
```

Variante : `script4a.sh`

On reprend la fonction `checkdir` qui lit la valeur de la variable donnée par l'utilisateur :

```
#!/bin/bash
# script4a.sh test si $passwdir existe
read -p "quel est le dossier à vérifier ?" passwdir
checkdir() {
    if [ -e $passwdir ]; then
        echo "le fichier $passwdir existe"
    else
        echo "le fichier $passwdir n'existe pas"
    fi
}
checkdir
exit
```

2.3. Structure de base d'un script

Quel serait la structure de base d'un script Bash ?

1. Shebang
2. Commentaires
3. Fonction gestion de la syntaxe
4. Fonction(s) utile(s)
5. Corps principal
6. Fin

```
#!/bin/bash
# script5.sh structure de base d'un script
target=$1
usage() {
    echo "Usage: $0 <fichier>"
    echo "Compte les lignes d'un fichier"
    exit
}
main() {
    ls -l $target
    echo "nombre de lignes : $(wc -l $target)"
    stat $target
}
if [ $# -lt 1 ]; then
    usage
elif [ $# -eq 1 ]; then
    main
else
    usage
fi
exit
```

2.4. Autres exemples de test

La page man de test pourrait nous inspirer, `man test`.

```
execverif() {
    if [ -x $target ] ; then
        #('x' comme "e_x_ecutable")
        echo $target " est exécutable."
    else
        echo $target " n'est pas exécutable."
    fi
}
```

```
#! /bin/sh
# 01_tmp.sh
dir="${HOME}/tmp/"
if [ -d ${dir} ] ; then
    rm -rf ${dir}
    echo "Le dossier de travail ${dir} existe et il est effacé"
fi
mkdir ${dir}
echo "Le dossier de travail ${dir} est créé"
```

3. Boucles

3.1. Boucle for-do

Faire la même chose pour tous les éléments d'une liste. En programmation, on est souvent amené à faire la même chose pour tous les éléments d'une liste. Dans un shell script, il est bien évidemment possible de ne pas

réécrire dix fois la même chose. On dira que l'on fait une boucle. Dans la boucle "for-do-done", la variable prendra successivement les valeurs dans la liste et les commandes à l'intérieur du "do-done" seront répétées pour chacune de ces valeurs.

```
for variable in liste_de_valeur ; do
    commande
    commande
done
```

Par défaut, `for` utilise la liste `in "$@"` si on omet ce mot-clé.

Supposons que nous souhaitions créer 10 fichiers .tar.gz factices, en une seule ligne :

```
for num in 0 1 2 3 4 5 6 7 8 9 ; do touch fichier$num.tar.gz ; done
```

Mieux :

```
for num in {0..9} ; do touch fichier$num.tar.gz ; done
```

Supposons que nous souhaitions renommer tous nos fichiers `*.tar.gz` en `*.tar.gz.old` :

```
#!/bin/bash
# script6.sh boucle
#x prend chacune des valeurs possibles correspondant au motif : *.tar.gz
for x in ./*.tar.gz ; do
    # tous les fichiers $x sont renommés $x.old
    echo "$x -> $x.old"
    mv "$x" "$x.old"
#on finit notre boucle
done
exit
```

Script inverse

Voici le script inverse, c'est sans compter sur de meilleurs outils dédiés à la maniplation des noms de fichier :

```
#!/bin/sh
# script6r.sh inverse
#x prend chacune des valeurs possibles correspondant au motif : *.tar.gz.old
for x in ./*.tar.gz.old ; do
    # tous les fichiers $x sont renommés $x sans le .old
    echo "$x -> ${x%.old}"
    mv $x ${x%.old}
# on finit notre boucle
done
exit
```

3.2. Boucle while

Faire une même chose tant qu'une certaine condition est remplie. Pour faire une certaine chose tant qu'une condition est remplie, on utilise une boucle de type "while-do-done" et "until-do-done".

```
while condition ; do
    commandes
done
```

`while ;do` répète les commandes tant que la condition est vérifiée.

```
until condition ; do
    commandes
done
```

`until ; do ; done` répète les commandes jusqu'à ce que la condition soit vraie, ou alors tant qu'elle est fausse.

Comment rompre ou reprendre une boucle ?

- Rupture avec `break`,
- Reprise avec `continue`.

Exercice.

Supposons, par exemple que vous souhaitiez afficher les 100 premiers nombres (pour une obscure raison) ou que vous vouliez créer 100 machines virtuelles.

```
#!/bin/bash
# script7.sh boucle while
i=0
while [ $i -lt 100 ] ; do
    echo $i
    i=$[$i+1]
done
exit
```

De manière plus élégante avec l'instruction `for` :

```
#!/bin/bash
# for ((initial;condition;action))
for ((i=0;i<100;i=i+1)); do
    echo $i
done
exit
```

3.3. Boucle case-esac

L'instruction "case-esac" permet de modifier le déroulement du script selon la valeur d'un paramètre ou d'une variable. On l'utilise le plus souvent quand les valeurs possibles sont en nombre restreint et peuvent être prévues. Les imprévus peuvent alors être représentés par le signe $*$.

Demandons par exemple à l'utilisateur s'il souhaite afficher ou non les fichiers cachés du répertoire en cours.

```
#!/bin/sh
# script8.sh case-esac
#pose la question et récupère la réponse
echo "Le contenu du répertoire courant va être affiché."
read -p "Souhaitez-vous afficher aussi les fichiers cachés (oui/non) : " reponse
#agit selon la réponse
case $reponse in
    oui)
        clear
        ls -a ;;
    non)
        ls;;
    *) echo "Veuillez répondre par oui ou par non." ;;
esac
exit
```

3.4. Divers

Boîtes de dialogue

On pourrait aussi s'intéresser à Whiptail : https ://en.wikibooks.org/wiki/Bash_Shell_Scripting/Whiptail qui permet de créer des boîtes de dialogue.

Déboggage de script

On peut débogguer l'exécution du script en le lançant avec bash -x. Par exemple :

```
$ bash -x script7.sh
```

Etude de ~/.bashrc

Le fichier `~/.bashrc` est lu à chaque connexion de l'utilisateur.

```
$ head ~/.bashrc
# .bashrc
# Source global definitions
if [ -f /etc/bashrc ]; then
    . /etc/bashrc
fi
# Uncomment the following line if you don't like systemctl's auto-paging feature:
# export SYSTEMD_PAGER=
```

4. Variables : concepts avancés

4.1. Affection des variables

- On affecte une valeur à une variable en la déclarant `variable=valeur`
- La valeur d'une variable est traitée par défaut comme une chaîne de caractère.
- Le nom d'une variable ne peut pas commencer par un chiffre.

4.2. Protection des variables

On peut annuler la signification des caractères spéciaux comme `*`, `?`, `#`, `|`, `[]`, `{}` en utilisant des caractères d'échappement, qui sont également des caractères génériques.

\ Antislash

L'antislash `\`, qu'on appelle le caractère d'échappement, annule le sens de tous les caractères génériques, en forçant le shell à les interpréter littéralement.

```
echo \$var
$var
echo "\$var"
$var
```

" " Guillemets

Les guillemets (doubles) `" "` sont les guillemets faibles mais annulent la plupart des méta-caractères entourés à l'exception du tube (`|`), de l'antislash (`\`) et des variables (`$var`).

```
var=5
echo la valeur de la variable est $var
la valeur de la variable est 5
echo "la valeur de la variable est $var"
la valeur de la variable est 5
```

' ' Apostrophes

Les guillemets simples, ou apostrophes (`' '`) annulent le sens de tous les caractères génériques sauf l'antislash.

```
echo '\$var'
\$var
```

4.3. Variables d'environnement

Variable shell $PS1

Le shell utilise toute une série de variables par exemple `$PS1` (Prompt String 1) :

```
echo $PS1
\[\e]0;\u@\h: \w\a\]${debian_chroot:+($debian_chroot)}\u@\h:\w\$
```

Cette variable liée à la session connectée est une variable d'environnement fixée dans le fichier `~/.bashrc`.

Variables d'environnement

Des variables d'environnement sont disponibles dans toutes les sessions. Par exemple PATH indique les chemins des exécutables :

```
echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games
```

Pour afficher les variables d'environnement :

```
printenv
```

4.4. Variables spéciales

- `$RANDOM` renvoie des valeurs aléatoires.
- Voir aussi https ://michel.mauny.net/sii/variables-shell.html
- Les variables suivantes sont relatives à la gestion des processus.

4.5. Portées des variables

Il y a deux types de variables : les variables locales et les variables globales (exportées).

Variables locales

Les variables locales ne sont accessibles que sur le shell actif. Les variables exportées ou globales sont accessibles à la fois par le shell actif et par tous les processus fils lancés à partir de ce shell.

- commande `set` / `unset`
- commande `env`
- commande `export` une variable, `export -f` pour une fonction
- préceder de `local` la valorisation d'une variable dans une fonction afin d'en limiter sa portée.

Variables globales

Commande `export`. La commande `export` rend la variable disponible dans tous les processus enfant de celui qui l'a lancée. Ainsi placée dans un fichier lu au démarrage d'une session "exportera" la valeur dans tous les shells "enfants".

4.6. Valeurs par défaut des variables

bash assign default value

Valeur par défaut -

```
${parameter-default}, ${parameter:-default}
```

Si le paramètre n'est pas défini, on utilise la valeur par défaut. Après l'appel, le paramètre n'est toujours pas défini. Le deux-points `:` ne fait une différence que lorsque le paramètre a été déclaré et est nul.

```
unset EGGS
echo 1 ${EGGS-spam}    # 1 spam
echo 2 ${EGGS:-spam}   # 2 spam

EGGS=
echo 3 ${EGGS-spam}    # 3
echo 4 ${EGGS:-spam}   # 4 spam

EGGS=cheese
echo 5 ${EGGS-spam}    # 5 cheese
echo 6 ${EGGS:-spam}   # 6 cheese
```

Valeur par défaut =

```
${parameter=default}, ${parameter:=default}
```

Si le paramètre n'est pas défini, on utilise la valeur par défaut. Les deux formes sont presque équivalentes. Le deux-points : ne fait une différence que lorsque le paramètre a été déclaré et est nul.

```
# sets variable without needing to reassign
# colons suppress attempting to run the string
unset EGGS
: ${EGGS=spam}
echo 1 $EGGS     # 1 spam
unset EGGS
: ${EGGS:=spam}
echo 2 $EGGS     # 2 spam

EGGS=
: ${EGGS=spam}
echo 3 $EGGS     # 3        (set, but blank -> leaves alone)
EGGS=
: ${EGGS:=spam}
echo 4 $EGGS     # 4 spam

EGGS=cheese
: ${EGGS:=spam}
echo 5 $EGGS     # 5 cheese
EGGS=cheese
: ${EGGS=spam}
echo 6 $EGGS     # 6 cheese
```

Valeur par défaut +

```
${parameter+alt_value}, ${parameter:+alt_value}
```

Si le paramètre est défini, on utilise `alt_value`, sinon on utilise une chaîne de caractères nulle. Après l'appel, la valeur du paramètre n'est pas modifiée. Le deux-points : ne fait une différence que lorsque le paramètre a été déclaré et est nul.

```
unset EGGS
echo 1 ${EGGS+spam}  # 1
echo 2 ${EGGS:+spam} # 2

EGGS=
echo 3 ${EGGS+spam}  # 3 spam
echo 4 ${EGGS:+spam} # 4

EGGS=cheese
echo 5 ${EGGS+spam}  # 5 spam
echo 6 ${EGGS:+spam} # 6 spam
```

4.7. Expansions de paramètres avec extraction

```
CHEMIN="/home/francois/archives/francois_2021-05-24-120409.zip"
```

Extraction de sous-chaînes

On peut extraire des sous-chaînes de caractères :

À partir du début de la valeur de la variable selon la méthode suivante `${variable :debut :longueur}`

```
echo ${CHEMIN}
/home/francois/archives/francois_2021-05-24-120409.zip
```

```
echo ${CHEMIN:16:7}
rchives
```

Recherche de motifs

Les caractères génériques englobent d'autres caractères :

- `*` signifie tout caractère
- `?` signifie un seul caractère
- `[Aa-Zz]` correspond à une plage
- `{home,zip}` corresond à une liste

Extraction du début et de la fin

Extraction du début retirant un motif selon `${variable#motif}` :

```
echo ${CHEMIN}
/home/francois/archives/francois_2021-05-24-120409.zip
```

```
echo ${CHEMIN#/home/francois}
/archives/francois_2021-05-24-120409.zip
```

```
echo ${CHEMIN#*francois}
/archives/francois_2021-05-24-120409.zip


echo ${CHEMIN##*francois}
_2021-05-24-120409.zip
```

Extraction de la fin

Extraction de la fin retirant un motif selon `${variable%motif}`

```
echo ${CHEMIN}
/home/francois/archives/francois_2021-05-24-120409.zip


echo ${CHEMIN%francois_2021-05-24-120409.zip}
/home/francois/archives/


echo ${CHEMIN%/francois*}
/home/francois/archives


echo ${CHEMIN%%/francois*}
/home
```

Remplacement sur motif

```
${variable/motif/remplacement}

echo ${CHEMIN/francois/amina}
/home/amina/archives/francois_2021-05-24-120409.zip


echo ${CHEMIN//francois/amina}
/home/amina/archives/amina_2021-05-24-120409.zip


echo ${CHEMIN//.zip/}
/home/amina/archives/amina_2021-05-24-120409
```

Compter les lettres

```
var=anticonstitutionnellement
echo Il y a ${#var} caractères dans cette variable
Il y a 25 caractères dans cette variable
```

Exercice de manipulation de variable

Considérons une liste de chemin de fichiers séparés par le signe `,` :

```
FICHIERS="/home/amina/francois/francois_2021-05-24-120409.zip,/home/amina/archives/amina_20\
21-05-22-220411.zip"
```

Exercice 1

Créer une liste de noms de fichier séparée par un espace avec cette variable.

```
francois_2021-05-24-120409.zip
amina_2021-05-22-220411.zip
```

Solution 1

```
for file_path in ${FICHIERS//,/ } ; do echo ${file_path##*/} ; done
```

Exercice 2

Extraire uniquement l'horodatage.

```
2021-05-24-120409
2021-05-22-220411
```

Solution 2

```
for file_path in ${FICHIERS//,/ } ; do
  name_extension=${file_path##*/} ; filename=${name_extension/.zip/}
  echo ${filename##*_}
done
```

4.8. Paramètres positionnels

Les paramètres positionnels représentent les éléments d'une commande en variables

On peut utiliser le script suivant pour illustrer les paramètres positionnels :

```
#! /bin/sh
# 06_affiche_arguments.sh
echo 0 : $0
if [ -n "$1"  ] ; then echo 1  : $1  ; fi
if [ -n "$2"  ] ; then echo 2  : $2  ; fi
if [ -n "$3"  ] ; then echo 3  : $3  ; fi
if [ -n "$4"  ] ; then echo 4  : $4  ; fi
if [ -n "$5"  ] ; then echo 5  : $5  ; fi
if [ -n "$6"  ] ; then echo 6  : $6  ; fi
if [ -n "$7"  ] ; then echo 7  : $7  ; fi
if [ -n "$8"  ] ; then echo 8  : $8  ; fi
if [ -n "$9"  ] ; then echo 9  : $9  ; fi
if [ -n "${10}" ] ; then echo 10 : ${10} ; fi
```

On obtient ceci :

```
./06_affiche_arguments.sh un deux trois quatre zozo petzouille sept huit neuf 10
```

```
0 : ./06_affiche_arguments.sh
1 : un
2 : deux
3 : trois
4 : quatre
5 : zozo
6 : petzouille
7 : sept
8 : huit
9 : neuf
10 : 10
```

4.9. Commande shift

On peut optimiser les opérations avec la commande `shift` qui décale les paramètres vers la gauche (supprime le premier paramètre) :

```
#! /bin/sh
# 07_affiche_arguments_3.sh
while [ -n "$1" ] ; do
  echo $1
  shift
done
```

`$#`représente le nombre total de paramètres. On peut voir ceci :

```
#! /bin/sh
# 08_affiche_arguments_4.sh
while [ $# -ne 0 ]; do
  echo $1
  shift
done
```

On peut encore illustrer d'autres paramètres positionnels :

```
#!/bin/bash
# 09_affiche_arguments_spéciaux.sh
echo "Nom du script $0"
echo "Premier paramètre $1"
echo "Second paramètre $2"
echo "Tous les paramètres $*"
echo "Tous les paramètres (préservant des espaces) $@"
echo "Nombre de paramètres $#"
echo "PID du shell {$$}"
echo "code de retour $?"
exit
```

4.10. Substitution de commandes

Le résultat d'une commande peut valoriser une variable :

```
kernel=$(uname -r)
echo $kernel
```

Ou encore selon cette méthode :

```
kernel=`uname -r`
echo $kernel
```

4.11. Expansions arithmétiques

```
$(( expression ))
```

```
declare -i variable
```

...

4.12. Tableaux

Exemple de création de tableau `var` :

```
var=('valeur1' 'valeur2' 'valeur3')
# ou
declare -a var=('valeur1' 'valeur2' 'valeur3')
# ou
var('valeur1' 'valeur2' 'valeur3')
# ou
var=([0]'valeur1' [1]'valeur2' [2]'valeur3')
```

Manipulations de base d'un tableau :

```
# Affiche toutes les entrées du tableau
echo ${var[@]}
valeur1 valeur2 valeur3
# Affiche toutes les entrées du tableau aussi
echo ${var[*]}
valeur1 valeur2 valeur3
# Affiche la valeur de l'indice 0 (premier)
echo ${var[0]}
valeur1
# Affiche le nombre d'indices
echo ${#var[@]}
3
Affiche tous les indices
echo ${!var[@]}
0 1 2
```

Bouclage dans un tableau :

```
for i in "${!var[@]}"; do
  printf "%s\t%s\n" "$i" "${var[$i]}"
done
```

Autres exemples, exercices et références

- The Bash Manual
- Introduction aux tableaux en bash
- Manipulation de tableaux indicés en bash
- Manipulation des chaînes de caractères et des tableaux en bash
- Chaînes et tableaux en BASH TP3
- script bash : Tableaux

4.13. Gestion des processus

```
# Create a process
cat /dev/urandom > /dev/null &
# Get the pid and write it somewhere
custompid=$! ; echo $custompid > /tmp/custompid.pid
# Check the process running
ps aux | grep random
# Kill the process with his pid
kill -9 `cat /tmp/custompid.pid`
# Clean the pid file
echo /dev/null > /tmp/custompid.pid
# Check if the process is running
ps aux | grep random
```

5. Modèles et figures Bash

5.1. Sélection d'instructions

Structure if-then-else

```
if condition_1
then
    commande_1
elif condition_2
then
    commande_2
else
    commande_n
fi
```

```
if condition ; then
    commande
fi
```

Conditions et tests

- La condition peut-être n'importe quelle commande,
- souvent la commande `test` représentée aussi par `[ ]` ou `[[ ]]`.
- Le code de retour est alors vérifié :
  - `0` : condition vraie
  - `1` : condition fausse

`man test` donne les arguments de la commande test.

Structure case-esac

```
case expression in
    motif_1 ) commande_1 ;;
    motif_2 ) commande_2 ;;
esac
```

L'expression indiquée à la suite du `case` est évaluée et son résultat est comparé aux différents motifs. En cas de correspondance avec le motif, une commande suivante est réalisée. Elle se termine par `;;`

Le motif peut comprendre des caractères génériques :

```
case
    *) ;;
    ?) ;;
    O* | o* | Y* | y*) ;;
    3.*) ;;
esac
```

Exercices

1. Écrivez un script qui vérifie l'existence d'au moins un paramètre dans la commande.
2. Écrivez un script qui vérifie que deux paramètres sont compris endéans un intervalle compris entre 0 et 100.
3. Écrivez un script qui demande O/o/Oui/oui et N/n/Non/non dans toutes ses formes et qui rend la valeur.
4. Écrivez un script qui ajoute un utilisateur existant dans un groupe.
5. Écrivez un script qui crée un rapport sommaire sur les connexions erronées sur votre machine.
6. Écrivez un script qui utilise les options de la commande `test` (ou `[ ]`) pour décrire les fichiers qui lui sont passés en argument.

5.2. Figures de boucles

```
for i in 0 1 2 3 ; do echo "ligne ${i}" ; done
```

```
for i in {0..3} ; do echo "ligne ${i}" ; done
```

```
i=0 ; while [ i < 4 ] ; do echo "ligne ${i}" ; i=$[$i+1] ; done
```

```
for ((i=0;i<4;i=i+1)); do echo "ligne ${i}" ; done
```

5.3. Figures de substitution

```
I="ubuntu2004.qcow2"
echo ${I#ubuntu2004.}
qcow2


I="ubuntu2004.qcow2"
echo ${I#*.}
qcow2


I="ubuntu2004.qcow2"
echo ${I%.qcow2}
ubuntu2004


I="ubuntu2004.qcow2"
echo ${I%.qcow2}
ubuntu2004


I="ubuntu2004.qcow2"
echo ${I:11}
qcow2


I="ubuntu2004.qcow2"
echo ${I/qcow2/img}
ubuntu2004.img


echo ${I/u/\-}
-buntu2004.qcow2


echo ${I//u/\-}
-b-nt-2004.qcow2
```

5.4. Figures de vérification

1. Fonction are_you_sure

```
are_you_sure () {
read -r -p "Are you sure? [y/N] " response
case "$response" in
    [yY][eE][sS]|[yY])
       sleep 1
        ;;
    *)
        exit
        ;;
esac
}
```

2. Fonction check_distribution

```
check_distribution () {
if [ -f /etc/debian_version ]; then
echo "Debian/Ubuntu OS Type"
elif [ -f /etc/redhat-release ]; then
echo "RHEL/Centos OS Type"
fi
}
```

3. Fonctions check_variable

```
check_variable () {
case ${variable} in
    isolated) echo "isolated" ;;
    nat) echo "nat" ;;
    full) echo "full"  ;;
    *) echo "isolated, nat or full ? exit" ; exit 1 ;;
esac
}
```

4. Fonction check_parameters

```
parameters=$#
check_parameters () {
# Check the number of parameters given and display help
if [ "$parameters" -ne 2  ] ; then
echo "Description : This script do this"
echo "Usage       : $0 <type : isolated or nat or full>"
echo "Example     : '$0 isolated' or '$0 nat'"
exit
fi
}
```

5. Fonction check_root_id

```
check_root_id () {
if [ "$EUID" -ne 0 ]
  then echo "Please run as root"
  exit
fi
}
```

6. Vérification de la disponibilité d'un binaire

```
curl -V >/dev/null 2>&1 || { echo >&2 "Please install curl"; exit 2; }
```

7. Tests avec grep et exécutions conditionnelles

```
if ! grep -q "vmx" /proc/cpuinfo ; then echo "Please enable virtualization instructions" ; \
exit 1 ; fi


{ grep -q "vmx" /proc/cpuinfo ; [ $? == 0 ]; } || { echo "Please enable virtualization inst\
ructions" ; exit 1 ;  }


[ `grep -c "vmx" /proc/cpuinfo` == 0 ] && { echo "Please enable virtualization instructions\
" ; exit 1 ;  }
```

8. Fonction check_interface

```
check_interface () {
if grep -qw "${interface:=lo}" <<< $(ls /sys/class/net) ; then
echo "This interface ${interface} exists"
else
echo "This interface ${interface} does not exist"
fi
}
```

5.5. Figures de génération aléatoire

1. Fonctions create_ip_range

```
net_id1="$(shuf -i 0-255 -n 1)"
net_id2="$(shuf -i 0-255 -n 1)"
# random /24 in 10.0.0.0/8 range
ip4="10.${net_id1}.${net_id2}."
ip6="fd00:${net_id1}:${net_id2}::"
# Fix your own range
#ip4="192.168.1."
#ip6="fd00:1::"
create_ip_range () {
# Reporting Function about IPv4 and IPv6 configuration
cat << EOF > ~/report.txt
```

```
Bridge IPv4 address : ${ip4}1/24
IPv4 range          : ${ip4}0 255.255.255.0
DHCP range          : ${ip4}128 - ${ip4}150
Bridge IPv6 address : ${ip6}1/64
IPv6 range          : ${ip6}/64
DHCPv6 range        : ${ip6}128/64 - ${ip6}150/64
DNS Servers         : ${ip4}1 and ${ip6}1
EOF
echo "~/report.txt writed : "
cat ~/report.txt
}
```

2.Fonction create_mac_address

```
create_mac_address () {
mac=$(tr -dc a-f0-9 < /dev/urandom | head -c 10 | sed -r 's/(..)/\1:/g;s/:$//;s/^/02:/')
echo $mac
}
```

3. Fonction de génération d'aléas / UUID

```
alea () {
apt-get -y install uuid-runtime openssl
alea1=$(< /dev/urandom tr -dc _A-Z-a-z-0-9 | head -c${1:-32};echo;)
echo "1. urandom alea : $alea1"
alea2=$(date +%s | sha256sum | base64 | head -c 32 ; echo)
echo "2. date alea $alea2"
alea3=$(openssl rand -base64 32)
echo "3. openssl alea : $alea3"
alea4=$(uuidgen -t)
echo "4. time based uuid : $alea4"
alea5=$(uuidgen -r)
echo "5. random based uuid : $alea5"
echo "6. random based uuid résumé : ${alea5:25}"
echo "7. random based uuid résumé : ${alea5//\-/}"
}
```

5.8. Getopts : arguments de la ligne de commande

`getopts` est outil puissant analyse les arguments de la ligne de commande transmis au script. C'est l'équivalent en Bash de la commande externe `getopt` et de la fonction de bibliothèque `getopt` familière aux programmeurs C. Il permet de passer et de concaténer plusieurs options et les arguments associés à un script (par exemple `scriptname -abc -e /usr/local`).

La construction `getopts` utilise deux variables implicites. `$OPTIND` est le pointeur d'argument ("OPTion INDex") et `$OPTARG` ("OPTion ARGument") l'argument (facultatif) attaché à une option. Un deux-points suivant le nom de l'option dans la déclaration marque cette option comme ayant un argument associé.

Une construction `getopts` est généralement empaquetée dans une boucle "while", qui traite les options et les arguments un par un, puis incrémente la variable implicite `$OPTIND` pour pointer vers la suivante.

Les arguments passés de la ligne de commande au script doivent être précédés d'un tiret (-). C'est le préfixe - qui permet à `getopts` de reconnaître les arguments de la ligne de commande comme des options. En fait, `getopts` ne traitera pas les arguments sans le préfixe -, et terminera le traitement de l'option au premier argument rencontré sans eux.

Le modèle `getopts` diffère légèrement du modèle standard en boucle, en ce sens qu'il ne contient pas de crochets de condition.

Exemple 1

```
#!/bin/bash
NO_ARGS=0
if [ $# -eq "$NO_ARGS" ]
then
  echo "Usage: `basename $0` options (-abc)"
  exit 1
fi

while getopts ":abc:" Option
do
  case $Option in
    a ) echo "option a [OPTIND=${OPTIND}]";;
    b ) echo "option b [OPTIND=${OPTIND}]";;
    c ) echo "option c [OPTIND=${OPTIND}] ${OPTARG}";;
  esac
done

shift $(($OPTIND - 1))
```

Exemple 2

An example of how to use getopts in bash

```
#!/bin/bash

usage() { echo "Usage: $0 [-s <45|90>] [-p <string>]" 1>&2; exit 1; }

while getopts ":s:p:" o; do
    case "${o}" in
        s)
            s=${OPTARG}
            ((s == 45 || s == 90)) || usage
            ;;
        p)
            p=${OPTARG}
            ;;
        *)
            usage
            ;;
    esac
done
shift $((OPTIND-1))
```

```
if [ -z "${s}" ] || [ -z "${p}" ]; then
    usage
fi

echo "s = ${s}"
echo "p = ${p}"
```

Démonstation :

```
$ ./myscript.sh
Usage: ./myscript.sh [-s <45|90>] [-p <string>]

$ ./myscript.sh -h
Usage: ./myscript.sh [-s <45|90>] [-p <string>]

$ ./myscript.sh -s "" -p ""
Usage: ./myscript.sh [-s <45|90>] [-p <string>]

$ ./myscript.sh -s 10 -p foo
Usage: ./myscript.sh [-s <45|90>] [-p <string>]

$ ./myscript.sh -s 45 -p foo
s = 45
p = foo

$ ./myscript.sh -s 90 -p bar
s = 90
p = bar
```

Exemple 3

How do I parse command line arguments in Bash ?

```
cat >/tmp/demo-getopts.sh <<'EOF'
#!/bin/sh

# A POSIX variable
OPTIND=1         # Reset in case getopts has been used previously in the shell.

# Initialize our own variables:
output_file=""
verbose=0

while getopts "h?vf:" opt; do
    case "$opt" in
    h|\?)
        show_help
        exit 0
        ;;
    v)  verbose=1
```

```
        ;;
    f)  output_file=$OPTARG
        ;;
    esac
done

shift $((OPTIND-1))

[ "${1:-}" = "--" ] && shift

echo "verbose=$verbose, output_file='$output_file', Leftovers: $@"
EOF

chmod +x /tmp/demo-getopts.sh

/tmp/demo-getopts.sh -vf /etc/hosts foo bar
```

Résultat :

```
verbose=1, output_file='/etc/hosts', Leftovers: foo bar
```

5.7. Modèle de script bash

Source : https ://github.com/leonteale/pentestpackage/blob/master/BashScriptTemplate.sh

```
#!/bin/bash
#################################################################################
# Copyright: Leon Teale @leonteale https://leonteale.co.uk
#################################################################################
#################################################################################
# Program: <APPLICATION DESCRIPTION HERE>
#################################################################################
VERSION="0.0.1"; # <release>.<major change>.<minor change>
PROGNAME="<APPLICATION NAME>";
AUTHOR="You, you lucky so and so";

#################################################################################
## Pipeline:
## TODO:
#################################################################################

#################################################################################
# XXX: Coloured variables
#################################################################################
red=`echo -e "\033[31m"`
lcyan=`echo -e "\033[36m"`
yellow=`echo -e "\033[33m"`
green=`echo -e "\033[32m"`
blue=`echo -e "\033[34m"`
purple=`echo -e "\033[35m"`
normal=`echo -e "\033[m"`
```

```
################################################################################
# XXX: Configuration
################################################################################

declare -A EXIT_CODES

EXIT_CODES['unknown']=-1
EXIT_CODES['ok']=0
EXIT_CODES['generic']=1
EXIT_CODES['limit']=3
EXIT_CODES['missing']=5
EXIT_CODES['failure']=10

DEBUG=0
param=""

################################################################################
# XXX: Help Functions
################################################################################
show_usage() {
        echo -e """Web Application scanner using an array of different pre-made tools\n
        Usage: $0 <target>
        \t-h\t\tshows this help menu
        \t-v\t\tshows the version number and other misc info
        \t-D\t\tdisplays more verbose output for debugging purposes"""

        exit 1
        exit ${EXIT_CODES['ok']};
}

show_version() {
        echo "$PROGNAME version: $VERSION ($AUTHOR)";
        exit ${EXIT_CODES['ok']};
}

debug() {
        # Only print when in DEBUG mode
        if [[ $DEBUG == 1 ]]; then
                echo $1;
        fi
}

err() {
        echo "$@" 1>&2;
        exit ${EXIT_CODES['generic']};
}

################################################################################
# XXX: Initialisation and menu
################################################################################
if [ $# == 0 ] ; then
        show_usage;
```

```
fi

while getopts :vhx opt
do
  case $opt in
  v) show_version;;
  h) show_usage;;
  *)  echo "Unknown Option: -$OPTARG" >&2; exit 1;;
  esac
done

# Make sure we have all the parameters we need (if you need to force any parameters)
#if [[ -z "$param" ]]; then
#        err "This is a required parameter";
#fi

################################################################################
# XXX: Kick off
################################################################################
header() {
        clear
        echo -e """
---------------------------------
 $PROGNAME v$VERSION $AUTHOR
---------------------------------\n"""
}

main() {

#start coding here
  echo "start coding here"

}

header
main "$@"

debug $param;
```

6. Script rm amélioré

Cette section est une reprise de l'exercice de script `rm_secure.sh` de Christophe Blaess.

On trouvera bon nombre d'exemples de scripts à télécharger sur la page https ://www.blaess.fr/christophe/livres/scripts-shell-linux-et-unix/. Le script `rm_secure.sh` est situé dans le dossier `exemples/ch02-Programmation_Shell/`.

6.1. Commande rm

```
rm --help
Usage: rm [OPTION]... FILE...
Remove (unlink) the FILE(s).

  -f, --force           ignore nonexistent files and arguments, never prompt
  -i                    prompt before every removal
  -I                    prompt once before removing more than three files, or
                          when removing recursively; less intrusive than -i,
                          while still giving protection against most mistakes
      --interactive[=WHEN]  prompt according to WHEN: never, once (-I), or
                          always (-i); without WHEN, prompt always
      --one-file-system  when removing a hierarchy recursively, skip any
                          directory that is on a file system different from
                          that of the corresponding command line argument
      --no-preserve-root  do not treat '/' specially
      --preserve-root   do not remove '/' (default)
  -r, -R, --recursive   remove directories and their contents recursively
  -d, --dir             remove empty directories
  -v, --verbose         explain what is being done
      --help     display this help and exit
      --version  output version information and exit

By default, rm does not remove directories.  Use the --recursive (-r or -R)
option to remove each listed directory, too, along with all of its contents.

To remove a file whose name starts with a '-', for example '-foo',
use one of these commands:
  rm -- -foo

  rm ./-foo
```

6.2. Description

- Il s'agit d'une fonction à "sourcer" qui ajoute des fonctionnalités à la commande `/bin/rm` : une sorte de corbeille temporaire
- Trois options supplémentaires et sept standards sont à interpréter
- Des fichiers/dossiers sont à interpréter comme arguments possibles
- Les fichiers/dossiers effacés sont placés dans une corbeille temporaire avant suppression.
- Ces fichiers peuvent être listés et restaurés à l'endroit de l'exécution de la commande.

Quelles options peut-on ajouter ?

- une vérification des droits sur le dossier temporaire
- une option qui précise le point de restauration (approche par défaut, récupération emplacement original absolu)
- une gestion des écrasements lors de la restauration (versionning, diff)
- une gestion des écrasements de fichiers mis en corbeille

6.3. Concepts

Le script met en oeuvre les notions suivantes :

- définition de variables
- imbrications de boucles
- boucle `while ; do command ; done`
- Traitement d'options `getopts`
- boucle `case/esac ) ;;`
- condition `if/then`
- tests `[ ]`
- trap commande signal

6.4. Structure

Le Script exécute un traitement séquentiel :

1. Déclarations de variables dont locales
2. Traitement des options
3. ...

6.5. Sourcer le script

En Bash :

```
source rm_secure.sh
```

6.6. Script automatique

Pour que le script démarre automatiquement au démarrage de la session de l'utilisateur :

- `~/.bashrc`
- `~/.bash_profile`

7. Références

- Wiki Bash Hackers
- Scripts shell Linux et Unix de Christophe Blaess
- https ://mywiki.wooledge.org/BashGuide
- Synatxe Bash obsolète et dépréciée
- Erreur du débutant en Bash
- Advanced Bash-Scripting Guide
- https ://bash.cyberciti.biz/guide/Main_Page
- Shell Style Guide

Archive d'exemples

Archive : Exercices de scripts sur les noms de fichiers

On vous présente un cas où nous sommes invités à renommer des fichiers ayant l'extension tar.gz en tar.gz.old et inversément.

Pour réaliser cet exercice nous avons besoin d'un certain nombre de fichiers. Une idée serait d'utiliser la commande `touch`. Supposons qu'il faille créer 100 fichiers numérotés dans un dossier temporaire.

Cas : vider et créer un dossier temporaire de travail

Pour vider et créer un dossier temporaire de travail, on pourrait proposer ceci d'illustrer la fonction conditionnelle `if condition ; then commandes ; else commandes ; fi` :

```
#! /bin/sh
# 01_tmp.sh
dir="${HOME}/tmp/"
if [ -d ${dir} ] ; then
    rm -rf ${dir}
    echo "Le dossier de travail ${dir} existe et il est effacé"
fi
mkdir ${dir}
echo "Le dossier de travail ${dir} est créé"
```

Cas : créer des fichiers à la volée

Pour créer des fichiers, on peut utilser la commande `touch` :

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

NAME
     touch -- change file access and modification times

SYNOPSIS
     touch [-A [-][[hh]mm]SS] [-acfhm] [-r file] [-t [[CC]YY]MMDDhhmm[.SS]] file ...

DESCRIPTION
     The touch utility sets the modification and access times of files.  If any file does n\
ot
     exist, it is created with default permissions.

     By default, touch changes both modification and access times.  The -a and -m flags may\
 be used
     to select the access time or the modification time individually.  Selecting both is eq\
uivalent
     to the default.  By default, the timestamps are set to the current time.  The -t flag \
explic-
     itly specifies a different time, and the -r flag specifies to set the times those of t\
he spec-
     ified file.  The -A flag adjusts the values by a specified amount.
```

Pour faire une certaine chose tant qu'une condition est remplie on utilise une boucle `while condition ; do commandes ; done`

```
#! /bin/sh
# 02_creation_fichiers0.sh
dir="${HOME}/tmp/"
i=0
while [ $i -lt 100 ] ; do
        touch ${dir}fic$i.tar.gz
        echo "Création de ${dir}fic$i.tar.gz"
        i=$[$i+1]
done
```

De manière peut-être plus élégante avec l'instruction `for ((initial ;condition ;action)) ; do commandes ; done` :

```
#!/bin/sh
# 03_creation_fichiers.sh
dir="${HOME}/tmp/"
i=0
#for ((initial;condition;action))
for ((i=0;i<100;i=i+1)); do
    touch ${dir}fic$i.tar.gz
        echo "Création de ${dir}fic$i.tar.gz"
done
```

Cas : renommage

Cas : renommage de *.tar.gz en *.tar.gz.old

Supposons maintenant que nous souhaitions renommer tous nos fichiers *.tar.gz en *.tar.gz.old, nous taperons le script suivant :

```
#!/bin/sh
# 04_renommage.sh
#x prend chacune des valeurs possibles correspondant au motif : *.tar.gz
dir="${HOME}/tmp/"
for x in ${dir}*.tar.gz ; do
  # tous les fichiers $x sont renommés $x.old
  echo "$x -> $x.old"
  mv "$x" "$x.old"
  # on finit notre boucle
done
```

Cas : renommage inverse

Cas : renommage inverse *.tar.gz.old *.gz.old

Voici le script inverse, c'est sans compter sur d'autres outils pour d'autres situations :

```
#!/bin/sh
# 05_denommage.sh
#x prend chacune des valeurs possibles correspondant au motif : *.tar.gz.old
dir="${HOME}/tmp/"
for x in ${dir}*.tar.gz.old ; do
  # tous les fichiers $x sont renommés $x sans le .old
  echo "$x -> ${x%.old}"
  mv $x ${x%.old}
  # on finit notre boucle
done
```

Cas : script extraction_serveurs.sh

On peut réaliser l'exercice `extraction_serveurs.sh`

Quatrième partie Virtualisation Linux

Objectifs des certification

RHCSA EX200 (RHEL7)

- **Utiliser des systèmes en cours d'exécution**
  - Accéder à la console d'une machine virtuelle
  - Démarrer et arrêter des machines virtuelles
- **Déployer, configurer et gérer des systèmes**
  - Installer Red Hat Enterprise Linux automatiquement à l'aide de Kickstart
  - Configurer une machine physique pour héberger des invités virtuels
  - Installer des systèmes Red Hat Enterprise Linux en tant qu'invités virtuels
  - Configurer des systèmes pour lancer des machines virtuelles au démarrage

LPIC 1

- *Sujet 102 : Installation de Linux et gestion de paquetages*
  - 102.6 Linux en tant que système virtuel hébergé

Introduction

Cette partie s'intéresse à la virtualisation Linux avec Qemu/KVM et la librairie Libvirt et différents outils de gestion Open Source. Un projet de scripts Bash prêt à l'emploi pour le déploiement et l'orchestration de machines virtuelles est proposé comme illustration.

4. Virtualisation KVM

On trouvera ici une initiation à la virtualisation native Linux Qemu/KVM exploitée avec la librairie Libvirt.

Introduction

Références à lire

- *RHEL 7 Virtualization Getting Started Guide, 2015*.
- *KVM Virtualization in RHEL 7 Made Easy, A Dell Technical White Paper*, September 2014.
- *RHEL7 Virtualization deployment and administration guide*, 2015.
- Openstack : DevStack
- KVM tools and enterprise usage

Scripts de préparation et d'automation

On trouvera sur le dépot GIT https ://github.com/goffinet/virt-scripts des scripts utiles à ce chapitre.

```
apt-get update && apt-get upgrade -y && apt-get install git -y
cd
git clone https://github.com/goffinet/virt-scripts
cd virt-scripts
```

Objectifs

1. Concepts virtualisation KVM
2. Installer KVM et ses outils de gestion
3. Créer une VM avec **`virt-manager`**
4. Administration avec **`virsh`**
5. Créer un dépôt local HTTP
6. Créer des VMs avec **`virt-install`**
7. Automatiser une l'installation avec **kickstart**
8. Accéder à la console (graphique te texte) et la dépanner

Marché de la virtualisation

Chaque Année Gartner publie une étude sur le marché global "Magic Quadrant for x86 Server Virtualization Infrastructure".

Selon Gartner en 2014, seulement 45% des systèmes Linux étaient virtualisés (contre 70% sur un OS concurrent). Par ailleurs, toujours selon la même source, la majorité des systèmes RHEL virtualisés fonctionnaient encore sous VMware ! La raison principale tiendrait à la difficulté de migrer les machines virtuelles vers un autre hyperviseur …

Toujours selon Gartner en 2015, la majorité des instances RHEL virtualisées sont exécutées sur VMware et VMware semblerait toujours aussi difficile à déplacer. Red Hat qui conduit le projet Open Source KVM dispose d'une véritable opportunité avec le développement de nouvelles infrastructures de type Cloud Privé avec Openstack malgré actuellement une forte préférence pour Ubuntu.

Selon Gartner en 2016, cette difficulté de migration est confirmée. Toutefois, on trouvera une concurrence forte dans les solutions basées Open Source entre Red Hat, Huawei, Sangfor, Virtuzzo (KVM) et Citrix, Oracle (Xen) et Ubuntu notamment autour d'OpenStack et des solutions de containers. Tous ces autres acteurs font concurrence contre les solutions Microsoft. A travers la *Linux Foundation* on trouve une force Open Source globalement concurrente.

1. Concepts

1.1. Terminologie

- Machine virtuelle ou “domaine invité” sont des ordinateurs dont le matériel est reproduit de manière logicielle sur l’hôte de virtualisation.
- L’hôte de virtualisation, machine physique *a priori*, embarque le logiciel hyperviseur qui interprète les pilotes de périphériques virtuels et offrent l’accès au processeur (CPU) et à la mémoire de travail (vive, RAM).
- Pour un déploiement en production, il est préférable d’équiper une infrastructure d’au moins deux hyperviseurs, un réseau de production, un réseau de gestion et un réseau de stockage avec un SAN dédié en iSCSI.
- Pour une solution de Lab, un PoC ou un projet personnel, il est préférable d’exécuter toutes fonctions sur un seul ordinateur quitte à passer à la *Nested Virtualization*.
- *Nested Virtualization* : La capacité de virtualiser un hypeviseur : KVM dans KVM, Hyper-V dans Hyper-V, VMWare ESXi dans ESXi, etc. Il est nécessaire d’activer les instructions “Intel VT-x or AMD-V” (32 bits) et Intel EPT or AMD RVI (64 bits) pour que l’hyperviseur virtualisé puisse accéder directement à ses instructions du CPU.

1.2. Typologie des architectures de virtualisation

- Isolateur : Docker, LXC, OpenVZ, BSD Jails
- Noyau en espace utilisateur
- Hyperviseur de type 2 / type 1
  - Virtualisation totale (Full virtualization) : qemu
  - Virtualisation Hardware-Assisted : qemu+KVM
  - Paravirtualisation : qemu+KVM+virtio, Xen

KVM est un hyperviseur de type 1 qui s’utilise aussi bien dans :

- Des environnements de développement, de test, d’apprentissage.
- La virtualisation de centres de données (data center).
- La mise en place d’infrastructures en nuage (cloud), il est l’hyperviseur par défaut sur OpenStack et beaucoup d’autres prestataires/solutions.

1.3. Machine virtuelle

Sur le plan technique, en général et particulièrement avec Libvirt, une machine virtuelle (VM) est représentée par :

1. un fichier de définition qui reprend les caractéristiques de la machine, par défaut situé (en format XML) dans `/etc/libvirt/qemu/`.
2. un ou des fichier(s) qui représentent les disques par défaut placés dans `/var/lib/libvirt/images/`.

Les principales ressources de virtualisation sont :

- La puissance (CPU/RAM)
- Le stockage (disques)
- Le réseau

Mais pour fonctionner, une VM a aussi besoin de bien d’autres interfaces matérielles qui peuvent être émulées ou para-virtualisées.

1.4. KVM

KVM Kernel-based Virtual Machine :

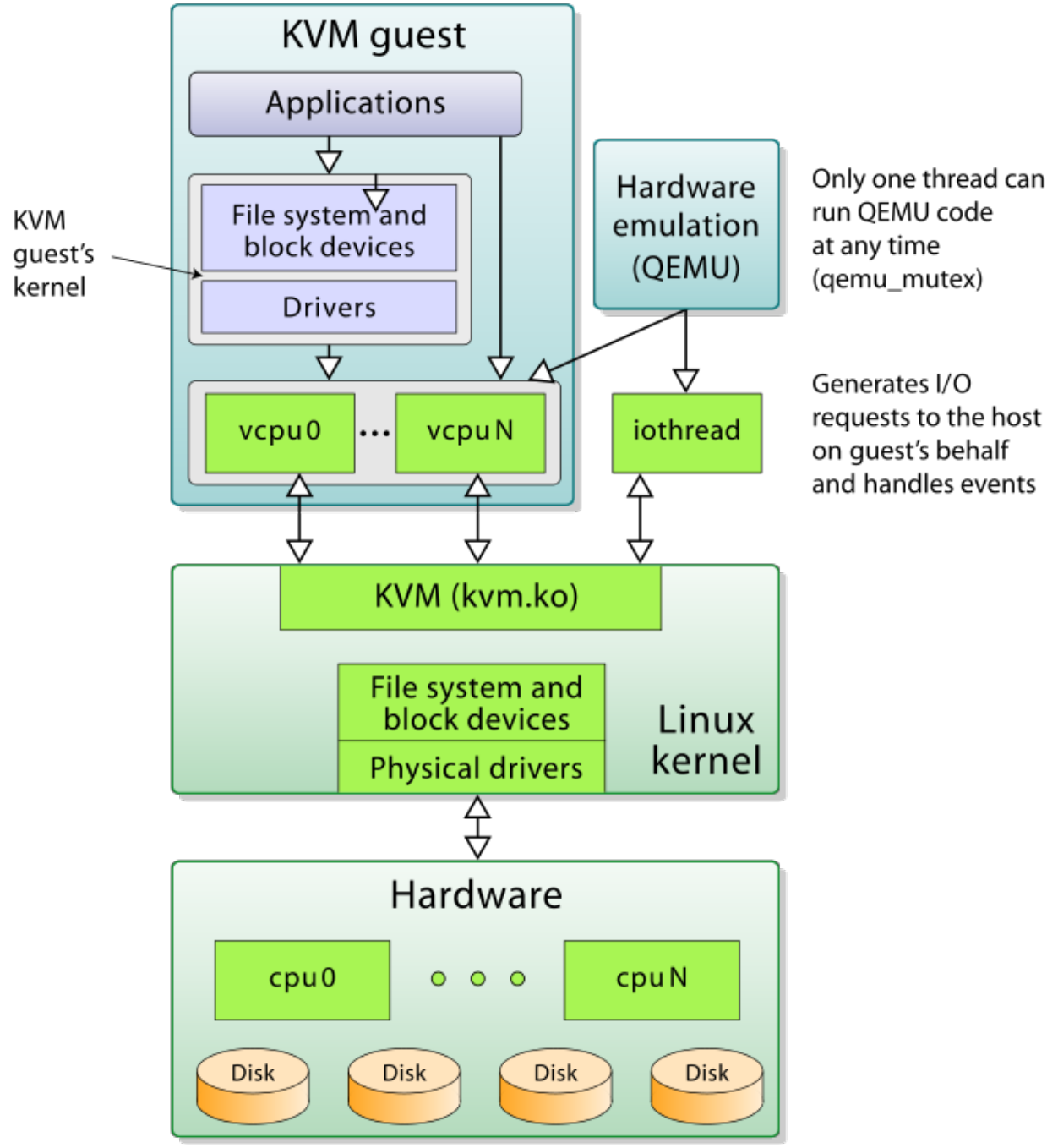
- KVM est le module qui transforme le noyau Linux en Hyperviseur type 1 (HVM et PV avec virtio). Ce module traduit rapidement les instructions des vCPU via les intructions VT AMD et Intel. Il prend aussi en charge des aspects de bas niveau de l'architecture x86.
- KVM est aussi un émulateur de matériel qui utilise **qemu** et les pilotes **virtio**.

Vue du noyau :

- Chaque VM est un processus
- Chaque vCPU est un thread de processeur.

Features :

- CPU and memory overcommit
- High performance paravirtual I/O
- Hotplug (cpu, block, nic)
- SMP guests
- Live Migration Power management
- PCI Device Assignment and SR-IOV
- KSM (Kernel Samepage Merging)
- SPICE, VNC, text
- NUMA



VM KVM

Source de l'image

1.5. Qemu

Qemu est un émulateur de diverses architectures dont x86 (Hyperviseur de type 2). Combiné au pilote KVM, il permet de réaliser de l'accélération Hardware (HVM).

L'outil de base `qemu-img` permet de créer et de gérer des images disque.

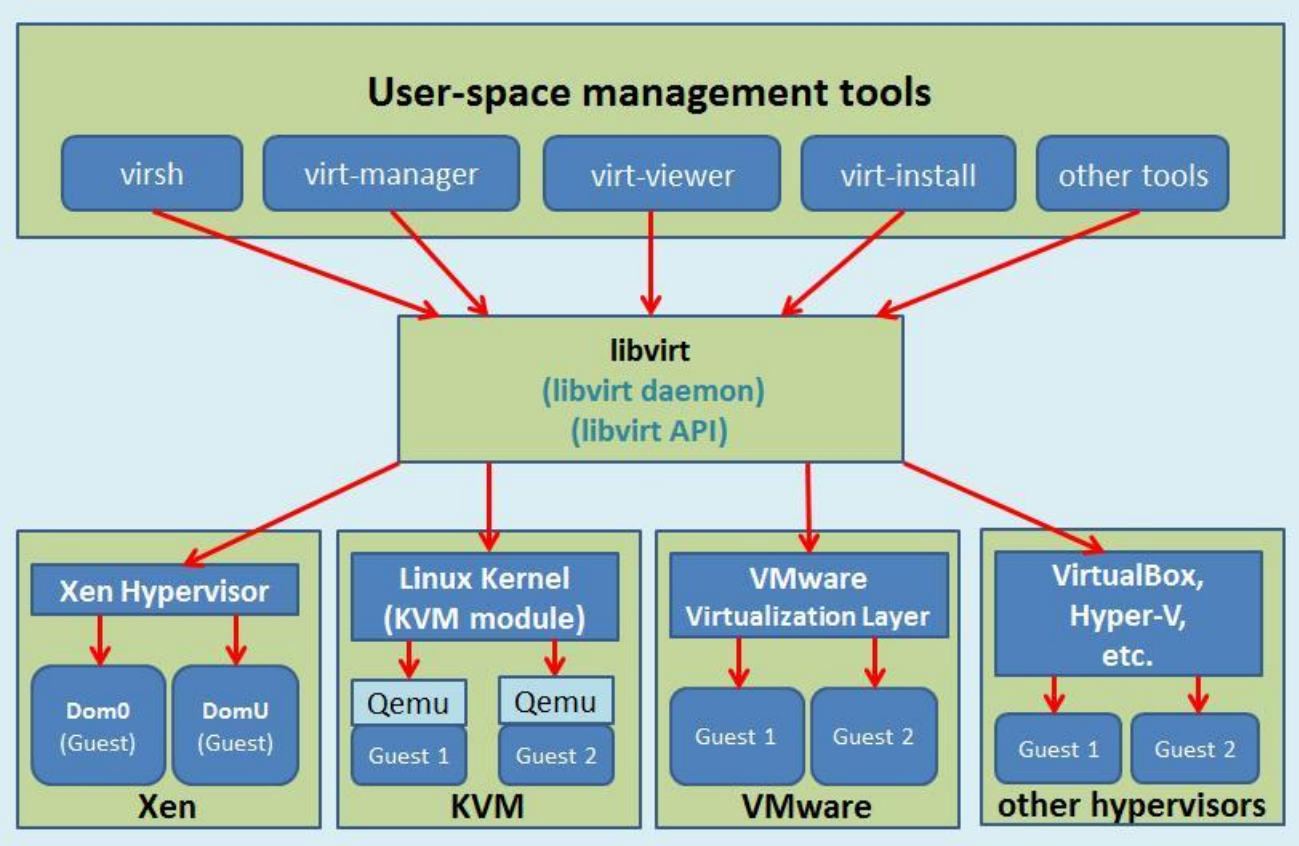
En format local les images disques peuvent se trouver en formats :

- raw
- qcow2

Par ailleurs, on peut utiliser directement des volumes logiques LVM.

1.6. Libvirt

libvirt un API de virtualisation Open Source qui s'interface avec un hyperviseur pour administrer les VMs.[1]



libvirt

1.7. Outils de base

- `virsh` : *cli* pour libvirt (ligne de commande et shell)
- `qemu-img` : permet de gérer les images des disques
- `virt-manager` : client graphique
- `virt-install` : commande pour la création des machines virtuelles
- `virt-viewer` : client console graphique (spice)
- `virt-clone` : outil de clonage
- `virt-top` : top de VM libvirt
- Autres outils

1.8. Outils libguestfs

`libguestfs` est un ensemble d'outils qui permettent d'accéder aux disques des machines virtuelles et de les modifier.

Ces outils offre pas mal de possibilités d'actions sur les diques virtuels :

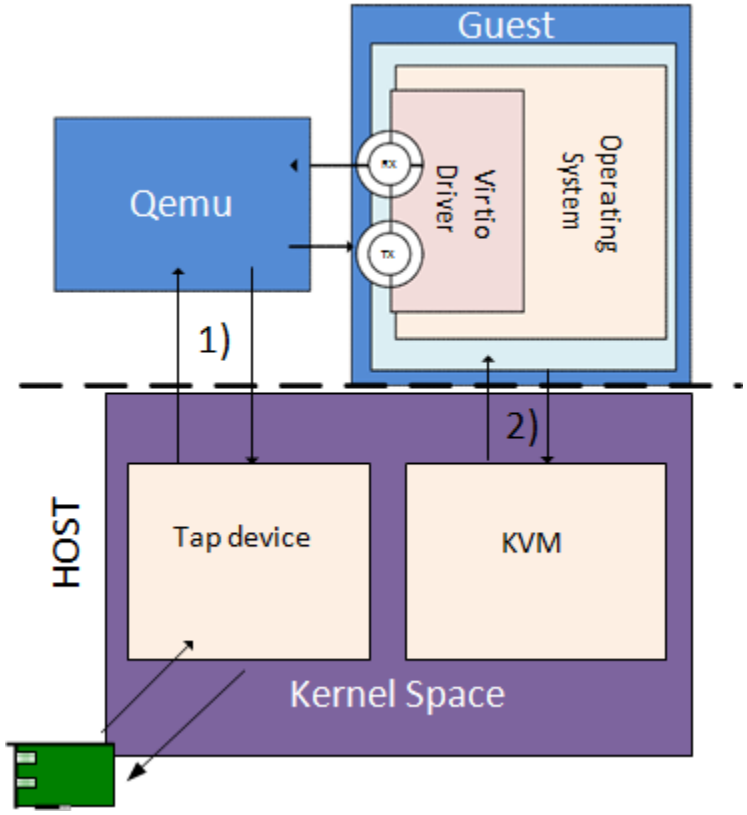
- accéder et de modifier un système de fichier invité à partir de l'hôte

1. (KVM⌧⌧) 6.1.1 libvirt⌧⌧

- `virt-builder` permet de créer des VM à partir d'un dépôt d'images
- `virt-sysprep`permet de "préparer" une VM à cloner
- obtenir des informations complètes dur l'usage des disques
- convertir des machines en P2V ou V2V
- ...

1.9. Pilotes et périphériques PV virtio

Périphérique réseau virtio. Ces périphériques totalement virtuels au plus proche du matériel disposent par défaut de leur pilotes disponibles dans tout noyau Linux récent et sont compatibles avec Microsoft Windows.



Périphérique réseau virtio

Source de l'image

1.10. Interfaces graphiques de gestion

- Kimchi est un outil de gestion en HTLM5 pour KVM basé sur libvirt. Il s'agit d'une solution à hôte unique.
- oVirt est aussi une plateforme Web de gestion de virtualisation multi-hôtes supportant d'autres hyperviseur, des volumes NFS, iSCSI ou FC (Fiber Channel), surveillance, fine tuning des ressources.

2. Installer KVM et ses outils de gestion

Les instructions VT doivent être activées dans le Bios :

```
grep -E 'svm|vmx' /proc/cpuinfo
```

Les deux valeurs correspondent aux capacités des processeurs :

- **vmx** : pour les processeurs Intel
- **svm** : pour les processeurs AMD

ou encore la commande `lscpu` donne cette information :

```
lscpu | grep Virtualisation
```

Mise à jour du sytème et installation des paquets KVM en RHEL7/Centos7.

```
yum update -y
yum group install "Virtualization Host" "Virtualization Client"

yum -y install \
qemu-kvm \
dejavu-lgc-sans-fonts \
libguestfs-tools
```

Démarrer le service libvirtd :

```
systemctl enable libvirtd && systemctl start libvirtd
```

Démarrer le service chronyd :

```
systemctl enable chronyd && systemctl start chronyd
```

Mise à jour du sytème et installation des paquets KVM en Debian 8.

```
apt-get update && sudo apt-get -y upgrade
apt-get -y install qemu-kvm libvirt-bin virtinst virt-viewer libguestfs-tools virt-manager \
uuid-runtime
```

Démarrage du commutateur virtuel par défaut.

```
virsh net-start default
virsh net-autostart default
```

Vérification du chargement du module kvm.

```
lsmod | grep kvm
```

Libvirt propose un outil de vérification de l'hôte.

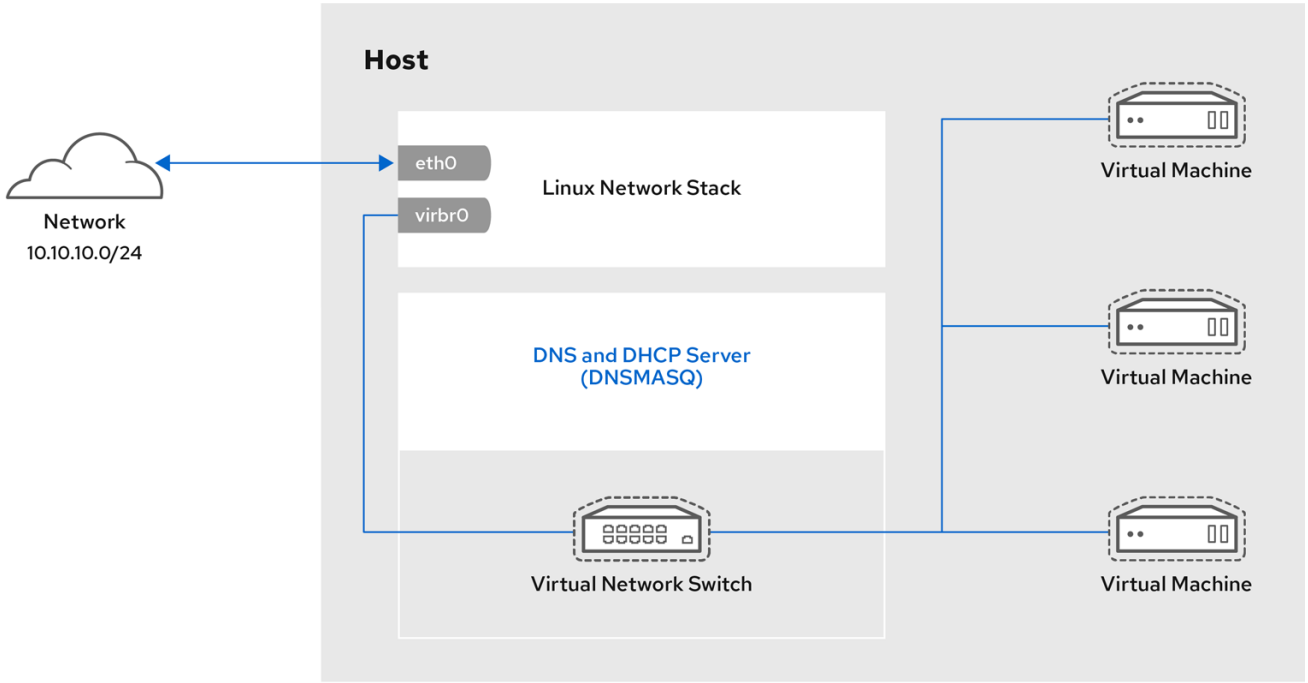
```
virt-host-validate
  QEMU: Vérification for hardware virtualization                 : PASS
  QEMU: Vérification for device /dev/kvm                         : PASS
  QEMU: Vérification for device /dev/vhost-net                   : PASS
  QEMU: Vérification for device /dev/net/tun                     : PASS
  LXC: Vérification pour Linux >= 2.6.26                        : PASS
```

Vérification du démarrage de libvirt :

```
systemctl status libvirtd
```

Configuration du réseau par défaut :

Une interface bridge `virbr0` 192.168.122.1 est "natée" à l'interface physique. Le démon `dnsmasq` fournit le service DNS/DHCP.



Source de l'image

```
ip add sh virbr0
ip route
iptables -t nat -L -n -v
cat /proc/sys/net/ipv4/ip_forward
```

L'emplacement par défaut de l'espace de stockage des disques est `/var/lib/libvirt/images/`. La définition des machines virtuelles est située dans `/etc/libvirt/qemu/`.

Il est peut-être plus aisé de désactiver pour l'instant firewalld (`systemctl stop firewalld`).

On propose ici un script de préparation de l'hôte de virtualisation :

https ://raw.githubusercontent.com/goffinet/virt-scripts/master/autoprep.sh

3. Création de VMs et administration de base

3.1. Créer une machine virtuelle avec virt-manager

Virt-manager est un outil graphique de gestion des hyperviseurs connecté via `libvirt`.

Dans une session X Window, suivre Quick Start with virt-manager ou encore KVM, Qemu, libvirt en images.

3.2. Administration de base avec virsh

A ajouter : images à exécuter

Avec `libvirt` et KVM, une “Machine Virtuelle (VM)” est appelée un “**Domaine**”.

Démarrage d’un domaine :

```
virsh start vm1
```

Arrêt d’un domaine :

```
virsh shutdown vm1
```

Extinction d’un domaine (comme on retire une prise de courant, il ne s’agit pas d’effacer le domaine) :

```
virsh destroy vm1
```

Pour retirer une VM de la gestion (le ou les disques associés persistent) :

```
virsh undefine vm1
```

Pour retirer un domaine de la gestion (et en effaçant ses disques) :

```
virsh undefine vm1 --remove-all-storage
```

Redémarrage d’un domaine :

```
virsh reboot vm1
```

Informations détaillées sur un domaine :

```
virsh dominfo vm1
```

Liste des domaines en fonction et éteints :

```
virsh list --all
```

Démarrage automatique du domaine au démarrage de l’hôte :

```
virsh autostart vm1
```

Désactiver l'activation au démarrage du domaine :

```
virsh autostart vm1 --disable
```

Accéder à la console série (texte) du domaine :

```
virsh console vm1
```

Accéder à la console graphique du domaine :

```
virt-viewer  vm1
```

4. Scripts d'installation

La commande qui permet de créer une machine virtuelle et de la lancer (pour y installer un système d'exploitation) est `virt-install`. Cette commande peut comporter un certain nombre de paramètres. Il est plus intéressant de travailler avec des scripts.

Dans le but de cloner un domaine existant, on s'intéressa à ce qui constitue fondamentalement une machine virtuelle :

- un fichier de définition de VM écrit en XML
- et un disque virtuel.

Les procédures de création ou de mise à jour d'objet (réseau, volume, domaine) avec la commande `virsh` consiste à manipuler des définitions XML :

- `define` / `undefine`
- `destroy` /`start` / `autostart`

4.1. Un premier script virt-install

On peut créer une machine virtuelle lancer une installation à partir du shell avec `virt-install` et des options.

Création et lancement d'une VM :

- RAM 1024/1 vCPU
- HD 8Go (raw)
- ttyS0
- console vnc
- Installation CD-ROM

```
#!/bin/bash
# vm-install1.sh

# local path to the iso
iso=/var/lib/iso/CentOS-7-x86_64-DVD-1611.iso

# Stop and undefine the VM
/bin/virsh destroy $1; /bin/virsh undefine $1 --remove-all-storage

# graphical console
# via local ISO
virt-install \
--virt-type kvm \
--name=$1 \
--disk path=/var/lib/libvirt/images/$1.img,size=8 \
--ram=1024 \
--vcpus=1 \
--os-variant=rhel7 \
--graphics vnc \
--console pty,target_type=serial \
--cdrom $iso
```

4.2. Export manuel d'une VM

Avant de procéder à un export, il est préférable de suspendre la VM d'origine :

```
virsh suspend vm1
```

On peut rediriger un "dump" de la VM d'origine dans un fichier xml.

```
virsh dumpxml vm1 > vm2.xml
```

Ensuite, il faut adapter ce fichier en retirant la valeur *id* et le champ *uuid* en modifiant le champ *name*, la balise *source file* qui désigne l'emplacement du nouveau disque, le champ *mac address* et en supprimant des balises entre </devices> et </domain>.

```
vi vm2.xml
```

Le second élément nécessaire à l'exécution de la VM est un disque dédié, soit la copie du disque de la machine originale :

```
cp /var/lib/libvirt/images/vm1.img /var/lib/libvirt/images/vm2.img
```

Enfin, on peut intégrer la machine à `libvirt`et la démarrer :

```
virsh define vm2.xml
virsh start vm2
```

A lire attentivement, voici le fichier `vm2.xml` adapté (uuid, devices, disks, mac) :

```
<domain type='kvm'>
  <name>vm2</name>
  <memory unit='KiB'>1048576</memory>
  <currentMemory unit='KiB'>1048576</currentMemory>
  <vcpu placement='static'>1</vcpu>
  <resource>
    <partition>/machine</partition>
  </resource>
  <os>
    <type arch='x86_64' machine='pc-i440fx-rhel7.0.0'>hvm</type>
    <boot dev='hd'/>
  </os>
  <features>
    <acpi/>
    <apic/>
    <pae/>
  </features>
  <cpu mode='custom' match='exact'>
    <model fallback='allow'>Westmere</model>
  </cpu>
  <clock offset='utc'>
    <timer name='rtc' tickpolicy='catchup'/>
    <timer name='pit' tickpolicy='delay'/>
    <timer name='hpet' present='no'/>
  </clock>
  <on_poweroff>destroy</on_poweroff>
  <on_reboot>restart</on_reboot>
  <on_crash>restart</on_crash>
  <devices>
    <emulator>/usr/libexec/qemu-kvm</emulator>
    <disk type='file' device='disk'>
      <driver name='qemu' type='raw'/>
      <source file='/var/lib/libvirt/images/vm2.img'/>
      <backingStore/>
      <target dev='vda' bus='virtio'/>
      <alias name='virtio-disk0'/>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x05' function='0x0'/>
    </disk>
    <controller type='usb' index='0' model='ich9-ehci1'>
      <alias name='usb0'/>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x04' function='0x7'/>
    </controller>
    <controller type='usb' index='0' model='ich9-uhci1'>
      <alias name='usb0'/>
      <master startport='0'/>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x04' function='0x0' multifuncti\
on='on'/>
    </controller>
    <controller type='usb' index='0' model='ich9-uhci2'>
      <alias name='usb0'/>
      <master startport='2'/>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x04' function='0x1'/>
    </controller>
```

```
    <controller type='usb' index='0' model='ich9-uhci3'>
      <alias name='usb0'/>
      <master startport='4'/>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x04' function='0x2'/>
    </controller>
    <controller type='pci' index='0' model='pci-root'>
      <alias name='pci.0'/>
    </controller>
    <interface type='bridge'>
      <mac address='52:54:00:8a:c3:2a'/>
      <source bridge='virbr0'/>
      <target dev='vnet0'/>
      <model type='virtio'/>
      <alias name='net0'/>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x03' function='0x0'/>
    </interface>
    <serial type='pty'>
      <source path='/dev/pts/0'/>
      <target type='isa-serial' port='0'/>
      <alias name='serial0'/>
    </serial>
    <console type='pty' tty='/dev/pts/0'>
      <source path='/dev/pts/0'/>
      <target type='serial' port='0'/>
      <alias name='serial0'/>
    </console>
    <input type='tablet' bus='usb'>
      <alias name='input0'/>
    </input>
    <memballoon model='virtio'>
      <alias name='balloon0'/>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x06' function='0x0'/>
    </memballoon>
  </devices>
</domain>
```

4.3. Clonage avec virt-clone

L'utilitaire **`virt-clone`** permet de cloner (à l'identique) une VM. `virt-clone`prend la peine de générer une nouvelle adresse MAC et un nouvel *uuid* pour le domaine. Il s'occupe également de dupliquer le ou les disques attachés au domaine.

Imaginons que l'on veuille cloner proprement la machine "vm1" en machine "vm2" dans le but de :

- générer une copie de sauvegarde de toute la machine,
- générer une copie de sauvegarde de base de donnée afin de travailler sur une base de donnée, hébergée (Dans ce cas, il est peut être intéressant de connecter le clone sur un réseau isolé du réseau de production.),
- dupliquer un même modèle.

  Attention, il faudra malgré tout modifier le fichier de configuration du réseau en retirant l'adresse mac et l'uuid dans le système d'exploitation invité. (`/etc/sysconfig/network-scripts/ifcfg-*`) si on désire exécuter le domaine original et son clone sur le même réseau ou si l'adresse IP est fixe.

Avant tout, vérifions l'état de la machine. Mettons-la en suspension pour assurer une copie correcte des disques :

```
virsh list
```

```
 ID    Nom                            État
----------------------------------------------------
 73    vm1                            en cours d'exécution
```

```
virsh suspend vm1
```

```
Domaine vm1 suspendu
```

```
virsh list
```

```
 ID    Nom                            État
----------------------------------------------------
 73    vm1                            mis en pause
```

Procédons au clonage :

```
virt-clone \
--original vm1 \
--name vm2 \
--file /var/lib/libvirt/images/vm2.img
```

```
virsh list --all
```

```
 ID    Nom                            État
----------------------------------------------------
 73    vm1                            mis en pause
 -     vm2                            fermé
```

Reprise de la machine :

```
virsh resume vm1
```

```
Domaine vm1 réactivé
```

4.4 Sysprep Linux

Pour la transformation d'une machine virtuelle en modèle (template), on peut utiliser `virt-sysprep` qui vient avec `libguestfs`, avant le clonage pour remettre à zéro ses propriétés.

```
virt-sysprep --list-operations
```

```
abrt-data * Remove the crash data generated by ABRT
bash-history * Remove the bash history in the guest
blkid-tab * Remove blkid tab in the guest
ca-certificates   Remove CA certificates in the guest
crash-data * Remove the crash data generated by kexec-tools
cron-spool * Remove user at-jobs and cron-jobs
customize * Customize the guest
dhcp-client-state * Remove DHCP client leases
dhcp-server-state * Remove DHCP server leases
dovecot-data * Remove Dovecot (mail server) data
firewall-rules   Remove the firewall rules
flag-reconfiguration   Flag the system for reconfiguration
fs-uuids   Change filesystem UUIDs
kerberos-data   Remove Kerberos data in the guest
logfiles * Remove many log files from the guest
lvm-uuids * Change LVM2 PV and VG UUIDs
machine-id * Remove the local machine ID
mail-spool * Remove email from the local mail spool directory
net-hostname * Remove HOSTNAME in network interface configuration
net-hwaddr * Remove HWADDR (hard-coded MAC address) configuration
pacct-log * Remove the process accounting log files
package-manager-cache * Remove package manager cache
pam-data * Remove the PAM data in the guest
puppet-data-log * Remove the data and log files of puppet
rh-subscription-manager * Remove the RH subscription manager files
rhn-systemid * Remove the RHN system ID
rpm-db * Remove host-specific RPM database files
samba-db-log * Remove the database and log files of Samba
script * Run arbitrary scripts against the guest
smolt-uuid * Remove the Smolt hardware UUID
ssh-hostkeys * Remove the SSH host keys in the guest
ssh-userdir * Remove ".ssh" directories in the guest
sssd-db-log * Remove the database and log files of sssd
tmp-files * Remove temporary files
udev-persistent-net * Remove udev persistent net rules
user-account   Remove the user accounts in the guest
utmp * Remove the utmp file
yum-uuid * Remove the yum UUID
```

5. Miroir d’installation HTTP

5.1. Miroir local

Les sources d’installation doivent contenir au minimum ceci :

```
{product path}
   |
   +--base
   |
   +--RPMS
```

product path :

- RedHat
- Fedora
- Centos

base : metadonnées

RPMS : Fichiers Red Hat Package Manager

Repo HTTP

Installer Apache :

```
yum -y install httpd
systemctl enable httpd.service && systemctl start httpd.service
```

Télécharger une image

```
mkdir -p /var/lib/iso
cd /var/lib/iso
wget https://centos.mirrors.ovh.net/ftp.centos.org/7/isos/x86_64/CentOS-7-x86_64-DVD-1611.i\
so
```

Monter l'ISO :

```
mount -o loop,ro CentOS*.iso /mnt
```

Copier les fichiers :

```
mkdir /var/www/html/repo/
cp -rp /mnt/* /var/www/html/repo/
chcon -R -t httpd_sys_content_t /var/www/html
```

Monter l'ISO directement dans `/var/www/html/repo/` est une alternative.

Miroirs publics externes

- OVH Centos 7
- Belnet Centos 7

5.2. Support d'installation HTTP

Création et lancement d'une VM :

- RAM 1024/1 vCPU
- HD 8Go (raw)
- ttyS0
- console vnc
- Installation repo HTTP local ou distant

```
#!/bin/bash
# vm-install2.sh

# KVM Host IP
bridge=192.168.122.1

# Repo URL
mirror=https://$bridge/repo
#mirror=https://centos.mirrors.ovh.net/ftp.centos.org/7/os/x86_64
#mirror=https://ftp.belnet.be/ftp.centos.org/7/os/x86_64
#mirror=https://mirror.i3d.net/pub/centos/7/os/x86_64

# Stop and undefine the VM
/bin/virsh destroy $1; /bin/virsh undefine $1 --remove-all-storage

# graphical console, bridged
# via http repo
virt-install \
--virt-type kvm \
--name=$1 \
--disk path=/var/lib/libvirt/images/$1.img,size=8 \
--ram=1024 \
--vcpus=1 \
--os-variant=rhel7 \
--network bridge=virbr0 \
--graphics vnc \
--console pty,target_type=serial \
--location $mirror
```

6. Installation automatique

6.1. Installation Kickstart

Kickstart permet d'automatiser les installations RHEL/Fedora/Centos (et d'autres) en indiquant un fichier de configuration qui est lu avant le logiciel d'installation Anaconda.

Documentation Kickstart

On rend le fichier kickstart disponible via HTTP (local, NFS, FTP) dans un dans dossier `conf` :

```
mkdir /var/www/html/conf/
touch vm.ks /var/www/html/conf/
chcon -R -t httpd_sys_content_t /var/www/html
```

Il est appelé par `virt-install` (directement au lancement du noyau). Ce fichier de configuration est rédigé automatiquement par Anaconda sur toute installation RHEL/Centos/Fedora :

```
less /root/anaconda-ks.cfg
```

On peut le générer en l'éditant via le programme `system-config-kickstart` :

```
yum install system-config-kickstart
system-config-kickstart
```

On peut toujours l'éditer manuellement.

6.2. Installation automatique en console graphique

Voici la configuration d'une installation simple avec un fichier `/var/www/html/conf/vm.ks` :

Clavier local, horodatage, réseau dhcp, nom, mot de passe, Swap, /boot, LVM, `@core`, chrony :‡

```
# File /var/www/html/conf/vm.ks

keyboard --vckeymap=be-oss --xlayouts='be (oss)'
lang fr_BE.UTF-8
network --onboot=on --bootproto=dhcp --device=link --hostname=localhost.localdomain
rootpw testtest
services --enabled="chronyd"
timezone Europe/Paris --isUtc
bootloader --location=mbr --boot-drive=vda
clearpart --all --initlabel --drives=vda
ignoredisk --only-use=vda
part pv.0 --fstype="lvmpv" --ondisk=vda --size=5000
part /boot --fstype="xfs" --ondisk=vda --size=500
volgroup vg0 --pesize=4096 pv.0
logvol swap  --fstype="swap" --size=500 --name=swap --vgname=vg0
logvol /  --fstype="xfs" --size=3072 --name=root --vgname=vg0

%packages --ignoremissing
@core
chrony
%end
reboot
```

Avec la machine virtuelle :

- RAM 1024/1 vCPU
- HD 8Go (qcow2)
- ttyS0
- console vnc
- Installation repo HTTP local ou distant

```
#!/bin/bash
# vm-install3.sh

bridge=192.168.122.1
mirror=https://$bridge/repo
#mirror=https://centos.mirrors.ovh.net/ftp.centos.org/7/os/x86_64
#mirror=https://ftp.belnet.be/ftp.centos.org/7/os/x86_64
#mirror=https://mirror.i3d.net/pub/centos/7/os/x86_64

#Stop and undefine the VM
/bin/virsh destroy $1; /bin/virsh undefine $1 --remove-all-storage

# Graphical console, bridged, HD qcow2
# HTTP + Kickstart
virt-install \
--virt-type kvm \
--name=$1 \
--disk path=/var/lib/libvirt/images/$1.qcow2,size=16,format=qcow2 \
--ram=1024 \
--vcpus=1 \
--os-variant=rhel7 \
--network bridge=virbr0 \
--graphics vnc \
--console pty,target_type=serial \
--location $mirror \
-x ks=https://$bridge/conf/vm.ks
```

6.3. Installation automatique en console texte

Voici la configuration d'une installation simple en console texte avec un fichier `/var/www/html/conf/vm2.ks`revu par `system-config-kickstart` :

Clavier local, horodatage, réseau dhcp, nom, firewall désactivé, selinux désactivé, pas de serveur X, console texte, mot de passe, Swap, /boot, LVM, @core, chrony :

```
# File /var/www/html/conf/vm2.ks
#platform=x86, AMD64, ou Intel EM64T
#version=DEVEL
# Install OS instead of upgrade
install
# Keyboard layouts
# old format: keyboard be-latin1
# new format:
keyboard --vckeymap=be-oss --xlayouts='be (oss)'
# Reboot after installation
reboot
# Root password
rootpw --plaintext testtest
# System timezone
timezone Europe/Paris
# System language
lang fr_BE
```

```
# Firewall configuration
firewall --disabled
# Network information
network  --bootproto=dhcp --device=link
# System authorization information
auth  --useshadow  --passalgo=sha512
# Use text mode install
text
# SELinux configuration
selinux --disabled
# Do not configure the X Window System
skipx

# System services
services --enabled="chronyd"

bootloader --location=mbr --boot-drive=vda
clearpart --all --initlabel --drives=vda
ignoredisk --only-use=vda
part pv.0 --fstype="lvmpv" --ondisk=vda --size=5000
part /boot --fstype="xfs" --ondisk=vda --size=500
volgroup vg0 --pesize=4096 pv.0
logvol swap  --fstype="swap" --size=500 --name=swap --vgname=vg0
logvol /  --fstype="xfs" --size=3072 --name=root --vgname=vg0

%packages --ignoremissing
@core
chrony

%end
```

Avec la machine virtuelle :

- RAM 1024/1 vCPU
- HD 8Go (qcow2)
- ttyS0
- console texte
- Installation repo HTTP local ou distant

```
#!/bin/bash
# vm-install4.sh

bridge=192.168.122.1
mirror=https://$bridge/repo
#mirror=https://centos.mirrors.ovh.net/ftp.centos.org/7/os/x86_64
#mirror=https://ftp.belnet.be/ftp.centos.org/7/os/x86_64
#mirror=https://mirror.i3d.net/pub/centos/7/os/x86_64

#Stop and undefine the VM
/bin/virsh destroy $1; /bin/virsh undefine $1 --remove-all-storage
```

```
# Text console, bridged, HD qcow2
# HTTP + Kickstart
virt-install \
--virt-type kvm \
--name=$1 \
--disk path=/var/lib/libvirt/images/$1.qcow2,size=16,format=qcow2 \
--ram=1024 \
--vcpus=1 \
--os-variant=rhel7 \
--network bridge=virbr0 \
--graphics none \
--console pty,target_type=serial \
--location $mirror \
-x "ks=https://$bridge/conf/vm2.ks console=ttyS0,115200n8 serial"
```

Pour échapper à la console texte :

```
CTRL+] (Linux)
CTRL+ALT+* (Mac OS X)
```

7. Accéder à la console

7.1. Accéder à la console graphique

Via une session Xwindows :

```
virt-manager
```

ou

```
virsh vncdisplay vm1
```

ou

```
netstat -tln|grep :59
vncviewer x.x.x.x:5900
```

7.2. Activer ttyS0 dans grub

Il faut nécessairement que la machine virtuelle dispose d'une console ttyS0 émulée !

Si la machine n'a pas été configurée avec ce paramètre grub, il faut editer le fichier **/etc/default/grub** en ajoutant `console=ttyS0`) à la variable `GRUB_CMDLINE_LINUX`. Le fichier `/etc/default/grub` devrait ressembler à ceci

```
cat /etc/default/grub
```

```
GRUB_TIMEOUT=5
GRUB_DISTRIBUTOR="$(sed 's, release .*$,,g' /etc/system-release)"
GRUB_DEFAULT=saved
GRUB_DISABLE_SUBMENU=true
GRUB_TERMINAL="serial console"
GRUB_SERIAL_COMMAND="serial --speed=115200"
GRUB_CMDLINE_LINUX="crashkernel=auto rd.lvm.lv=vg0/root rd.lvm.lv=vg0/swap console=ttyS0,11\
5200n8"
GRUB_DISABLE_RECOVERY="true"
```

Exécuter :

```
grub2-mkconfig -o /boot/grub2/grub.cfg
reboot
```

7.3. Accès à la console texte

```
virsh console vm1
```

```
Connected to domain vm1
Escape character is ^]

CentOS Linux 7 (Core)
Kernel 3.10.0-229.el7.x86_64 on an x86_64

localhost login:
```

8. Installation d’un invité MS-Windows

Afin de remplir le paramètres `--os-variant` de la commande `virt-install` vérifions les versions de Windows reconnues par la version actuelle de Libvirt :

```
osinfo-query os | grep Microsoft
```

A condition de disposer d’un ISO de Windows 7, voici un script de configuration d’une VM Windows en mode HVM :

```
#!/bin/bash
# File vm-install5.sh

#Stop and undefine the VM
/bin/virsh destroy $1; /bin/virsh undefine $1 --remove-all-storage

# Graphical console, bridged, cd-rom
virt-install \
--virt-type kvm \
--name=$1 \
--disk path=/var/lib/libvirt/images/$1.qcow2,size=16,format=qcow2 \
```

```
--cdrom /var/win7.iso \
--ram=2048 \
--vcpus=2 \
--arch=x86_64
--os-type=windows \
--os-variant=win7 \
--hvm \
--network bridge=virbr0 \
--keymap=fr \
--sound \
--vnc
```

On a aussi besoin des pilotes `virtio` si on utilise la paravirtualisation :

- Pilote de disque `bus=virtio,cache=none` ; d'autres bus disponibles tels que 'ide', 'sata', 'scsi', 'usb'.
- Pilote de carte réseau `model=virtio` ; d'autres options de modèle sont 'e1000' ou 'rtl8139'.

```
#!/bin/bash
# File vm-install6.sh
...
```

Voir Wiki sur les pilotes virtio :

```
wget https://fedorapeople.org/groups/virt/virtio-win/virtio-win.repo -O /etc/yum.repos.d/vi\
rtio-win.repo
yum install virtio-win
```

Où `/usr/share/virtio-win/*.iso` contient tous les pilotes virtio.

Après l'installation insérer l'iso virtio-tools, mais avant il est nécessaire de trouver le lecteur cd-rom :

```
virsh domblklist vm3
```

```
virsh change-media vm3 hdb /usr/share/virtio-win/virtio-win.iso
```

9. Manipulation de disques

9.1 Conversion de disques

raw vers qcow2

Pour convertir un disque **`raw`** vers **`qcow2`** :

```
#!/bin/bash
# File raw2qcow2.sh

path=/var/lib/libvirt/images/$1

virsh list
virsh suspend $1
ls -lh $path.img
echo "Conversion du disque"
qemu-img convert -c -O qcow2 $path.img $path.qcow2
mv $path.img $path.old
mv $path.qcow2 $path.img
virsh resume $1
ls -lh $path.*
```

vdi vers raw

Pour convertir un disque VB **`vdi`** vers **`raw`** :

```
VBoxManage clonehd --format RAW WindowsXP.vdi WindowsXP.raw


0%...10%...20%...30%...50%...70%...80%...90%...100%
Clone hard disk created in format 'RAW'. UUID: cfe44508-d957-4c8c-b5c5-2b4f266830d8
```

9.2. Redimensionnement de disques

...

9.3 Import d'une VM via son disque

Import d'une VM via son disque (converti d'une autre solution par exemple ou encore à partir d'une image de base) avec `virt-install --import --noautoconsole`.

```
#!/bin/bash
## This script import and launch minimal KVM images with a text console ##
## First download all the qcow2 images on https://get.goffinet.org/kvm/ ##
## Usage : bash define-guest.sh <name> <image> ##
## Reset root password with the procedure : ##
## https://linux.goffinet.org/processus_et_demarrage.html#10-password-recovery  ##
## Please check all the variables ##
# First parmater as name
name=$1
# Secund parameter image name avaible on "https://get.goffinet.org/kvm/"
# Image name : 'debian7', 'debian8', 'centos7', 'ubuntu1604', 'metasploitable', kali
image="$2.qcow2"
# Generate an unique string
uuid=$(uuidgen -t)
# VCPUs
vcpu="1"
```

```
# The new guest disk name
disk="${name}-${uuid:25}.qcow2"
# Diskbus can be 'ide', 'scsi', 'usb', 'virtio' or 'xen'
diskbus="virtio"
size="8"
# Hypervisor can be 'qemu', 'kvm' or 'xen'
hypervisor="kvm"
# RAM in Mb
memory="256"
# Graphics 'none' or 'vnc'
graphics="none"
# Network inetrface and model 'virtio' or 'rtl8139' or 'e1000'
interface="virbr0"
model="virtio"
# Parameters for metasploitable guests
if [ $image = "metasploitable.qcow2" ]; then
diskbus="scsi"
memory="512"
model="e1000"
fi
# Parameters for Kali guests
if [ $image = "kali.qcow2" ]; then
memory="1024"
size="16"
fi
## Local image copy to the default storage pool ##
cp ./$image /var/lib/libvirt/images/$disk
## Import and lauch the new guest ##
virt-install \
--virt-type $hypervisor \
--name=$name \
--disk path=/var/lib/libvirt/images/$disk,size=$size,format=qcow2,bus=$diskbus \
--ram=$memory \
--vcpus=$vcpu \
--os-variant=linux \
--network bridge=$interface,model=$model \
--graphics $graphics \
--console pty,target_type=serial \
--import \
--noautoconsole
```

9.4. Migration V2V

- Conversion V2V vers KVM

9.5. Manipulation de disques

Comment ajouter un disque vide à la volée sur un domaine invité ?

Ce script demande trois paramètres :

- `$1` : le nom de l'invité

- `$2` : le nom du disques
- `$3` : la taille du disque en Go

https ://raw.githubusercontent.com/goffinet/virt-scripts/master/add-storage.sh

```
#!/bin/bash
# Variables
guest=$1
disk=/var/lib/libvirt/images/${1}-${2}.img
size=$3
seek=$[${size}*1024]
# Create Spare Disk with dd
dd if=/dev/zero of=$disk  bs=1M seek=$seek count=0
# Or create a qcow2 disk
#qemu-img create -f qcow2 -o preallocation=metadata $disk ${size}G
# Attach the disk on live guest with persistence
virsh attach-disk $guest $disk $2 --cache none --live --persistent
# Detach the disk
#virsh detach-disk $guest $disk --persistent --live
```

Note.

- Manipulation de disques avec `kpartx`

10. Storage Pools / Storage Volumes

Ajouter un pool pour stocker les disques des domaines

```
virsh pool-define-as Images dir - - - - /home/so/Documents/kvm/images
virsh pool-list --all
virsh pool-build Images
virsh pool-start Images
virsh pool-autostart Images
virsh pool-list
virsh pool-info Images
```

10.1. Storage Pools

- LVM2
- iSCSI

11. Live Migration

- Live Migration

12. Réseau

12.1. Création d'un nouveau réseau virtuel

Lister les réseaux virtuels disponibles :

```
virsh net-list
```

Lister les adresses IP attribuées par un réseau virtuel :

```
virsh net-dhcp-leases default
 Expiry Time          MAC address        Protocol  IP address                Hostname      \
  Client ID or DUID
-------------------------------------------------------------------------------------------\
------------------------
 2017-02-16 16:52:03  52:54:00:15:35:f8  ipv4      192.168.122.129/24        arch          \
  ff:00:15:35:f8:00:01:00:01:20:33:48:d8:52:54:00:01:fb:0d
 2017-02-16 16:52:15  52:54:00:3d:6c:39  ipv4      192.168.122.126/24        u1            \
  -
 2017-02-16 16:52:11  52:54:00:50:ac:4a  ipv4      192.168.122.253/24        c1            \
  -
 2017-02-16 16:52:05  52:54:00:87:40:65  ipv4      192.168.122.130/24        d1            \
  -
 2017-02-16 16:52:09  52:54:00:91:31:a2  ipv4      192.168.122.212/24        k1            \
  -
 2017-02-16 16:52:20  52:54:00:b1:a7:a7  ipv4      192.168.122.173/24        m1            \
  -
```

Obtenir la base d'une définition de réseau virtuel (ici `default`) dans un fichier `lab.xml` :

```
virsh net-dumpxml default > lab.xml
```

La commande uuidgen permet de générer un uuid :

```
uuidgen
```

```
vim lab.xml
```

Si on définit un nouveau réseau "lab" utilisant l'interface virbr1 pour laquelle le NAT est activé, voici à quoi ce fichier devrait ressembler :

```
<network>
  <name>lab</name>
  <uuid>8293bf7a-ccf6-461b-8466-a058e7346d79</uuid>
  <forward mode='nat'>
    <nat>
      <port start='1024' end='65535'/>
    </nat>
  </forward>
  <bridge name='virbr1' stp='on' delay='0'/>
  <mac address='52:54:00:63:e8:10'/>
  <ip address='192.168.22.254' netmask='255.255.255.0'>
    <dhcp>
      <range start='192.168.22.100' end='192.168.22.150'/>
    </dhcp>
  </ip>
</network>
```

La balise *forward mode* peut prendre des valeurs comme *nat*, *route*, *bridge*, etc. En son absence le réseau est isolé.

Installation du nouveau réseau nommé "lab" :

```
virsh net-define lab.xml
```

Démarrage du réseau :

```
virsh net-start lab
```

Ensuite, faire en sorte qu'il démarre automatiquement :

```
virsh net-autostart lab
```

Pour attacher un domaine existant au réseau lab, il faut créer un fichier qui reprend les paramètres d'une interface :

```
vim virbr1.xml
```

```
<interface type='bridge'>
      <mac address='52:54:00:f7:e3:53'/>
      <source bridge='virbr1'/>
      <target dev='vnet0'/>
      <model type='virtio'/>
      <alias name='net0'/>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x03' function='0x0'/>
</interface>
```

Pour mettre à jour l'attachement de la carte réseau :

```
virsh update-device vm-test virbr1.xml
```

12.1. Ajout d'une seconde interface à un domaine

On peut prendre une définition d'interface comme suit. Il s'agit de modifier balise "alias name" en mettant une nouvelle valeur `net1` par exemple et en modifiant l'adresse MAC pour la rendre originale :

```
<interface type='bridge'>
      <mac address='52:54:00:aa:bb:cc'/>
      <source bridge='virbr1'/>
      <target dev='vnet0'/>
      <model type='virtio'/>
      <alias name='net0'/>
</interface>
```

On peut alors attacher la carte au domaine en "live" :

```
virsh attach-device nom_de_domaine fichier.xml
```

12.3. Réseau isolé

Création d'un réseau isolé nommé "lan" sur l'interface "virbr3"

https ://github.com/goffinet/virt-scripts/blob/master/add-isolated-bridge.sh

```
#!/bin/bash
# Create an isolated bridge

bridge="virbr3"
name=lan
path=/tmp
cat << EOF > $path/$name.xml
<network>
  <name>$name</name>
  <bridge name='$bridge' stp='on' delay='0'/>
</network>
EOF

virsh net-destroy $name
virsh net-create $path/$name.xml
#virsh net-autostart $name
```

12.4. Exercice : créer un routeur virtuel Linux

Voir chapitre sur le routage et le pare-feu :

1. Solution Routeur virtuel (libvirtd) interne sans DHCP
2. Solution KVM avec un routeur Centos/Debian Firewall
3. Solution KVM avec OpenWRT

13. Exemples de scripts automatiques

13.1. virt-builder

Le logiciel `virt-builder` permet de construire rapidement une VM à partir d'une image disponible sur le site de libguestfs. Il est paramétrable : voir https ://libguestfs.org/virt-builder.1.html.

```
virt-builder --list
```

On citera aussi le projet OZ : https ://github.com/clalancette/oz.

13.2. Exemples de code de déploiement

- CentOS-KVM-Image-Tools
- ostolc.org

14. Automation des installations

14.1. Améliorations des scripts précédents

On peut tenter d'améliorer les scripts des exercices précédents et se poser quelques questions

Configuration profils de VM

- Déploiement de template ou machines fraîchement installées ?
- Stockage capacité ?
- Stockage type ? qcow2 ou raw
- RAM minimum ?
- Réseau ponté, NAT ?

Configuration Kickstart

- adresse IP statique ou dhcp gérée ?
- IPv6
- partitionnement automatique / personnalisé LVM2
- hostname
- clé SSH
- paquets nécessaires
- commandes post install (service, hostname, fichiers, etc.)

14.2. Projet

Déploiement silencieux selon un profil de VM et d'installation.

Etude de cas :

- déploiement et gestion de VPS

Résumé

- profils de VM (Centos 7)
  - small, medium, large
  - modèle à cloner : small + virt-sysprep + sparsify + virt-clone
- profils d'installation
  - core + bootproto dhcp
  - docker-engine, httpd, mariadb, … + bootproto static
- Déploiement de services
  - Traditionnel : scripts et fichier Kickstart
  - Ansible : playbooks (modules rpm, …)
  - Docker
  - Une combinaison
- Surveillance / rapports
  - scripts
  - solutions commerciales / WebUI

Pré-requis

Cet exercice est réalisé sous Centos 7.

Un serveur Web sur l'hyperviseur, `httpd` par exemple, ou situé ailleurs rend disponible deux dossiers dans `/var/www/html` :

- `/var/www/html/repo` : contient la copie d'un CD d'installation
- `/var/www/html/conf` : contient le script de création, les fichiers Kickstart générés et une clé publique SSH

Deux sources sont à définir :

- Sources d'installation : **HTTP** ou local
- Fichier Kickstart : **HTTP** ou local

Profil de machine virtuelle “small”

Un script de création VM “small” :

- 1 vCPU
- 1 Go RAM
- 16 Go qcow2
- NIC : virb1 (192.168.22.0)

profil d’installation “core”

Configuration du système prédéfini dans un fichier Kickstart auquel correspond un profil d’installation Centos 7 avec un minimum de paquets, authentification à clé SSH et partionnement LVM2.

Il est nécessaire copier la clé publique de l’administrateur afin d’assurer

14.3. Première procédure

Firewalld désactivé

Pour les besoins de l’exercice, on désactivera le pare-feu.

```
systemctl stop firewalld
```

Création d’un réseau NAT dénommé lab

- NAT
- 192.168.22.254/24
- DHCP .100-.150
- virb1

Script virt-install “autovm.sh”

```
#!/bin/bash

## usage : autovm.sh [type] [nom domaine]
##         autovm.sh $1 $2
## types (appel d'un fichier kickstarts):
##         core

## Variables
## 1. $type : Fichier Kickstart
type=$1
## 2. $name : Nom du domaine
name=$2
## 3. $vol : Emplacement des disques
vol=/var/lib/libvirt/images
## 4. $conf : Emplacement HTTP des fichiers Kickstart
conf=https://192.168.122.1/conf
## 5. $mirror : Sources d'installation HTTP
mirror=https://192.168.122.1/repo
```

```
## Miroirs publics
#mirror=https://centos.mirrors.ovh.net/ftp.centos.org/7/os/x86_64
#mirror=https://ftp.belnet.be/ftp.centos.org/7/os/x86_64
#mirror=https://mirror.i3d.net/pub/centos/7/os/x86_64
## 6. $temp : nom temporaire pour le fichier Kickstart
temp="$name-$(uuidgen | cut -d - -f 1)"
## 7. $www : emplacement physique des fichiers de configuration
www=/var/www/html/conf

gest_dom ()
{
## Gestion des noms utilisés, ici pour un lab (à améliorer)
## Arrêt et retrait de la VM
echo "Arrêt et retrait de la VM $name"
/bin/virsh destroy $name; /bin/virsh undefine $name --remove-all-storage
}


prep_ks ()
{
## Préparation du fichier Kickstart
echo "Préparation du fichier Kickstart"
cp $www/$type.ks $www/$temp.ks
chown apache:apache $www/$temp.ks
## Report du hostname
sed -i "s/network --hostname=.*/network --hostname=$name/g" $www/$temp.ks
##
}

virt_install ()
{
## Démarrage de l'installation du domaine
echo "Démarrage de l'installation du domaine $name"
## Installation et lancement silencieux en mode texte
## selon la baseline définie Centos 7 1GB/1vCPU/HD8GB/1NIC/ttyS0
nohup \
virt-install \
--virt-type kvm \
--name=$name \
--disk path=/var/lib/libvirt/images/$name.qcow2,size=8,format=qcow2 \
--ram=1024 \
--vcpus=1 \
--os-variant=rhel7 \
--network bridge=virbr0 \
--graphics none \
--noreboot \
--console pty,target_type=serial \
--location $mirror \
-x "ks=$conf/$temp.ks console=ttyS0,115200n8 serial" \
> /dev/null 2>&1 &


## choix installation cdrom avec Kickstart local
```

```
#ks=/var/www/html/conf
#iso=path/to/iso
#--cdrom $iso \
#--initrd-inject=/$temp.ks -x "ks=file:/$temp.ks console=ttyS0,115200n8 serial" \
}

gest_dom
prep_ks
virt_install

# rm -f $www/$temp.ks
```

Fichier kickstart core.ks

```
## hostname
## bootproto dhcp ou static
##NAME            MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
## vda              252:0    0    8G  0 disk
## ├─vda1           252:1    0  500M  0 part /boot
## ├─vda2           252:2    0  6,7G  0 part        (auto-grow)
## │ └─local0-root 253:0    0  3,9G  0 lvm  /
## └─vda3           252:3    0  820M  0 part [SWAP] (auto-grow)
## install @core (minimum de paquets)
## post-install : update
## post-configuration ssh
install
keyboard --vckeymap=be-oss --xlayouts='be (oss)'
reboot
rootpw --plaintext testtest
timezone Europe/Brussels
url --url="https://192.168.122.1/repo"
lang fr_BE
firewall --disabled
network --bootproto=dhcp --device=eth0
network --hostname=template
# network --device=eth0 --bootproto=static --ip=192.168.22.10 --netmask 255.255.255.0 --gat\
eway 192.168.22.254 --nameserver=192.168.22.11 --ipv6 auto
auth  --useshadow  --passalgo=sha512
text
firstboot --enable
skipx
ignoredisk --only-use=vda
bootloader --location=mbr --boot-drive=vda
zerombr
clearpart --all --initlabel
part /boot --fstype="xfs" --ondisk=vda --size=500
part swap --recommended
part pv.00 --fstype="lvmpv" --ondisk=vda --size=500 --grow
volgroup local0 --pesize=4096 pv.00
logvol /  --fstype="xfs"  --size=4000 --name=root --vgname=local0
%packages
@core
```

```
%end
%post
#yum -y update
mkdir /root/.ssh
curl https://192.168.122.1/conf/id_rsa.pub > /root/.ssh/authorized_keys
sed -i 's/PasswordAuthentication yes/PasswordAuthentication no/g' /etc/ssh/sshd_config
%end
```

Le projet a evolué en des scripts plus succints, notamment avec le script https ://raw.githubusercontent.com/goffinet/virt-scripts/master/auto-install.sh :

```
#!/bin/bash

image=$1 # centos, debian, ubuntu
name=$2
fr_ubuntu_mirror=https://fr.archive.ubuntu.com/ubuntu/dists/xenial/main/installer-amd64/
fr_debian_mirror=https://ftp.debian.org/debian/dists/jessie/main/installer-amd64/
ovh_ubuntu_mirror=https://mirror.ovh.net/ubuntu/dists/xenial/main/installer-amd64/
ovh_debian_mirror=https://debian.mirrors.ovh.net/debian/dists/jessie/main/installer-amd64/
ovh_centos_mirror=https://centos.mirrors.ovh.net/ftp.centos.org/7/os/x86_64
belnet_ubuntu_mirror=https://ftp.belnet.be/ubuntu.com/ubuntu/dists/xenial/main/installer-am\
d64/
belnet_debian_mirror=https://ftp.belnet.be/debian/dists/jessie/main/installer-amd64/
belnet_centos_mirror=https://ftp.belnet.be/ftp.centos.org/7/os/x86_64
local_ubuntu_iso=/var/lib/iso/ubuntu-16.04.1-server-amd64.iso
url_ubuntu_iso=https://releases.ubuntu.com/16.04/ubuntu-16.04.1-server-amd64.iso
local_debian_iso=/var/lib/iso/debian-8.6.0-amd64-netinst.iso
url_debian_iso=https://cdimage.debian.org/debian-cd/8.6.0/amd64/iso-cd/debian-8.6.0-amd64-n\
etinst.iso
local_centos_iso=/var/lib/iso/CentOS-7-x86_64-DVD-1611.iso
url_centos_iso=https://ftp.belnet.be/ftp.centos.org/7/isos/x86_64/CentOS-7-x86_64-DVD-1611.\
iso
ubuntu_mirror=$belnet_ubuntu_mirror
debian_mirror=$fr_debian_mirror
centos_mirror=$belnet_centos_mirror

check_apache () {
yum install -y httpd curl || apt-get install apache2 curl
firewall-cmd --permanent --add-service=http
firewall-cmd --reload
systemctl enable httpd
systemctl start httpd
mkdir -p /var/www/html/conf
echo "this is ok" > /var/www/html/conf/ok
local check_value="this is ok"
local check_remote=$(curl -s https://127.0.0.1/conf/ok)
if [ "$check_remote"="$check_value" ] ; then
 echo "Apache is working"
else
 echo "Apache is not working"
 exit
fi
}
```

```
ubuntu_install () {
local url=https://192.168.122.1/conf/ubuntu1604-preseed.cfg
local mirror=$ubuntu_mirror

touch /var/www/html/conf/ubuntu1604-preseed.cfg
cat << EOF > /var/www/html/conf/ubuntu1604-preseed.cfg
d-i debian-installer/language                                string      en_US:en
d-i debian-installer/country                                 string      US
d-i debian-installer/locale                                  string      en_US
d-i debian-installer/splash                                  boolean     false
d-i localechooser/supported-locales                          multiselect en_US.UTF-8
d-i pkgsel/install-language-support                          boolean     true
d-i console-setup/ask_detect                                 boolean     false
d-i keyboard-configuration/modelcode                         string      pc105
d-i keyboard-configuration/layoutcode                        string      be
d-i debconf/language                                         string      en_US:en
d-i netcfg/choose_interface                                  select      auto
d-i netcfg/dhcp_timeout                                      string      5
d-i mirror/country                                           string      manual
d-i mirror/http/hostname                                     string      fr.archive.ubuntu.c\
om
d-i mirror/http/directory                                    string      /ubuntu
d-i mirror/http/proxy                                        string
d-i time/zone                                                string      Europe/Paris
d-i clock-setup/utc                                          boolean     true
d-i clock-setup/ntp                                          boolean     false
d-i passwd/root-login                                        boolean     false
d-i passwd/make-user                                         boolean     true
d-i passwd/user-fullname                                     string      user
d-i passwd/username                                          string      user
d-i passwd/user-password                                     password    testtest
d-i passwd/user-password-again                               password    testtest
d-i user-setup/allow-password-weak                           boolean     true
d-i passwd/user-default-groups                               string      adm cdrom dialout l\
padmin plugdev sambashare
d-i user-setup/encrypt-home                                  boolean     false
d-i apt-setup/restricted                                     boolean     true
d-i apt-setup/universe                                       boolean     true
d-i apt-setup/backports                                      boolean     true
d-i apt-setup/services-select                                multiselect security
d-i apt-setup/security_host                                  string      security.ubuntu.com
d-i apt-setup/security_path                                  string      /ubuntu
tasksel tasksel/first                                        multiselect openssh-server
d-i pkgsel/include                                           string      openssh-server pyth\
on-simplejson vim
d-i pkgsel/upgrade                                           select      safe-upgrade
d-i pkgsel/update-policy                                     select      none
d-i pkgsel/updatedb                                          boolean     true
d-i partman/confirm_write_new_label                          boolean     true
d-i partman/choose_partition                                 select      finish
d-i partman/confirm_nooverwrite                              boolean     true
d-i partman/confirm                                          boolean     true
```

```
d-i partman-auto/purge_lvm_from_device                  boolean     true
d-i partman-lvm/device_remove_lvm                       boolean     true
d-i partman-lvm/confirm                                 boolean     true
d-i partman-lvm/confirm_nooverwrite                     boolean     true
d-i partman-auto-lvm/no_boot                            boolean     true
d-i partman-md/device_remove_md                         boolean     true
d-i partman-md/confirm                                  boolean     true
d-i partman-md/confirm_nooverwrite                      boolean     true
d-i partman-auto/method                                 string      lvm
d-i partman-auto-lvm/guided_size                        string      max
d-i partman-partitioning/confirm_write_new_label        boolean     true
d-i grub-installer/only_debian                          boolean     true
d-i grub-installer/with_other_os                        boolean     true
d-i finish-install/reboot_in_progress                   note
d-i finish-install/keep-consoles                        boolean     false
d-i cdrom-detect/eject                                  boolean     true
d-i preseed/late_command in-target sed -i 's/PermitRootLogin\ prohibit-password/PermitRootL\
ogin\ yes/' /etc/ssh/sshd_config ; in-target wget https://gist.githubusercontent.com/goffin\
et/f515fb4c87f510d74165780cec78d62c/raw/7cf2c788c1c5600f7433d16f8f352c877a281a6a/ubuntu-gru\
b-console.sh ; in-target sh ubuntu-grub-console.sh
EOF

virt-install \
--virt-type kvm \
--name=$name \
--disk path=/var/lib/libvirt/images/$name.qcow2,size=8,format=qcow2 \
--ram=512 \
--vcpus=1 \
--os-variant=ubuntusaucy \
--network bridge=virbr0 \
--graphics none \
--console pty,target_type=serial \
--location $mirror \
-x "auto=true hostname=$name domain= url=$url text console=ttyS0,115200n8 serial"
}

debian_install () {
local url=https://192.168.122.1/conf/debian8-preseed.cfg
local mirror=$debian_mirror

touch /var/www/html/conf/debian8-preseed.cfg
cat << EOF > /var/www/html/conf/debian8-preseed.cfg
d-i debian-installer/locale string en_US
d-i keyboard-configuration/xkb-keymap select be
d-i netcfg/choose_interface select auto
d-i netcfg/get_hostname string unassigned-hostname
d-i netcfg/get_domain string unassigned-domain
d-i netcfg/wireless_wep string
d-i mirror/country string manual
d-i mirror/http/hostname string ftp.debian.org
d-i mirror/http/directory string /debian
d-i mirror/http/proxy string
d-i passwd/make-user boolean false
```

```
d-i passwd/root-password password testtest
d-i passwd/root-password-again password testtest
d-i clock-setup/utc boolean true
d-i time/zone string Europe/Paris
d-i clock-setup/ntp boolean true
d-i partman-auto/method string lvm
d-i partman-lvm/device_remove_lvm boolean true
d-i partman-md/device_remove_md boolean true
d-i partman-lvm/confirm boolean true
d-i partman-lvm/confirm_nooverwrite boolean true
d-i partman-auto/choose_recipe select atomic
d-i partman-partitioning/confirm_write_new_label boolean true
d-i partman/choose_partition select finish
d-i partman/confirm boolean true
d-i partman/confirm_nooverwrite boolean true
d-i partman-md/confirm boolean true
d-i partman-partitioning/confirm_write_new_label boolean true
d-i partman/choose_partition select finish
d-i partman/confirm boolean true
d-i partman/confirm_nooverwrite boolean true
tasksel tasksel/first multiselect standard
d-i pkgsel/include string openssh-server vim
d-i pkgsel/upgrade select full-upgrade
popularity-contest popularity-contest/participate boolean false
d-i grub-installer/only_debian boolean true
d-i grub-installer/with_other_os boolean true
d-i grub-installer/bootdev  string /dev/vda
d-i finish-install/keep-consoles boolean true
d-i finish-install/reboot_in_progress note
d-i preseed/late_command string in-target sed -i 's/PermitRootLogin\ without-password/Permi\
tRootLogin\ yes/' /etc/ssh/sshd_config; in-target wget https://gist.githubusercontent.com/g\
offinet/f515fb4c87f510d74165780cec78d62c/raw/7cf2c788c1c5600f7433d16f8f352c877a281a6a/ubunt\
u-grub-console.sh ; in-target sh ubuntu-grub-console.sh
EOF

virt-install \
--virt-type kvm \
--name=$name \
--disk path=/var/lib/libvirt/images/$name.qcow2,size=8,format=qcow2 \
--ram=512 \
--vcpus=1 \
--os-variant=debianwheezy \
--network bridge=virbr0 \
--graphics none \
--console pty,target_type=serial \
--location $mirror \
-x "auto=true hostname=$name domain= url=$url text console=ttyS0,115200n8 serial"
}

centos_install () {

local url=https://192.168.122.1/conf/centos7.ks
local mirror=$centos_mirror
```

```
read -r -d '' packages <<- EOM
@core
wget
EOM

touch /var/www/html/conf/centos7.ks
cat << EOF > /var/www/html/conf/centos7.ks
install
reboot
rootpw --plaintext testtest
keyboard --vckeymap=be-oss --xlayouts='be (oss)'
timezone Europe/Paris --isUtc
#timezone Europe/Brussels
lang en_US.UTF-8
#lang fr_BE
#cdrom
url --url="$mirror"
firewall --disabled
network --bootproto=dhcp --device=eth0
network --bootproto=dhcp --device=eth1
network --hostname=$name
# network --device=eth0 --bootproto=static --ip=192.168.22.10 --netmask 255.255.255.0 --gat\
eway $bridgeip4 --nameserver=$bridgeip4 --ipv6 auto
#auth  --useshadow  --passalgo=sha512
text
firstboot --enable
skipx
ignoredisk --only-use=vda
bootloader --location=mbr --boot-drive=vda
zerombr
clearpart --all --initlabel
#autopart --type=thinp # See the bug resolved in 7.3 https://bugzilla.redhat.com/show_bug.c\
gi?id=1290755
autopart --type=lvm
#part /boot --fstype="xfs" --ondisk=vda --size=500
#part swap --recommended
#part pv.00 --fstype="lvmpv" --ondisk=vda --size=500 --grow
#volgroup local0 --pesize=4096 pv.00
#logvol /  --fstype="xfs"  --size=4000 --name=root --vgname=local0
%packages
$packages
%end
%post
yum -y update && yum -y upgrade
#mkdir /root/.ssh
#curl ${conf}/id_rsa.pub > /root/.ssh/authorized_keys
#sed -i 's/PasswordAuthentication yes/PasswordAuthentication no/g' /etc/ssh/sshd_config
%end
EOF


virt-install \
```

```
--virt-type=kvm \
--name=$name \
--disk path=/var/lib/libvirt/images/$name.qcow2,size=8,format=qcow2 \
--ram=2048 \
--vcpus=1 \
--os-variant=rhel7 \
--network bridge=virbr0 \
--graphics none \
--noreboot \
--console pty,target_type=serial \
--location $mirror \
-x "auto=true hostname=$name domain= ks=$url text console=ttyS0,115200n8 serial"
}

start_install () {
if [ $image = centos ] ; then
 centos_install
elif [ $image = debian ] ; then
 debian_install
elif [ $image = ubuntu ] ; then
 ubuntu_install
else
 echo "Erreur dans le script : ./auto-install.sh [ centos | debian | ubuntu ] nom_de_vm"
 exit
fi
}

check_apache
start_install
```

14.4. Automation Ansible

Pré-requis

Seul pré-requis : Sytème Linux (Centos 7) avec un accès ssh avec authentification avec clé et Python installé.

Une résolution de nom robuste est conseillée.

Concepts

- Inventaire (fichier hosts)
- Modules
- Playbooks

Installation

```
yum install ansible
echo "nameserver 192.168.122.1" >> /etc/resolv.conf
mv /etc/ansible/hosts /etc/ansible/hosts.old
echo -e "[lab]\nvm0[1:4]" > /etc/ansible/hosts
cat /etc/ansible/hosts
```

Modules

```
ansible all -m ping
ansible vm01 -m ping
ansible all -m setup
ansible lab -m yum -a "name=openssh-server state=present"
```

Playbooks

A suivre dans un autre document.

14.5. Seconde procédure

Dans cette seconde procédure, on pourra choisir le profil de la machine virtuelle à partir d'un seul script.

Ce script vise à créer une machine virtuelle KVM d'une certaine capacité (small, medium, large) de manière automatique.

L'installation est minimale mais suffisante (core) pour assurer la gestion par Ansible et déployer des containers en tant que services.

On peut l'améliorer dans le profilage des installations (pré-installation, services, fichiers de configuration mais aussi dans la maintenance de la machine virtuelle (fin de l'installation, suppression du fichier Kickstart, –> via boucle while/surveillance du processus lancé, génération de rapports, logs) ou encore la gestion des erreurs.

Une option "gold image" ou modèle qui permettrait de préparer une VM et de cloner une telle installation fraîche serait un must, car elle répondrait à autre approche d'une solution de déploiement automatique de machines virtuelles.

Pour comprendre ce script, on le lira en trois temps :

1. variables générales au début
2. et leur corps principal (tout à la fin)
3. qui appelle trois fonctions :
    - gest_dom : qui efface le domaine invoqué (niveau de sévérité : lab)
    - prep_ks : qui prépare le fichier d'installation Kickstart
    - virt_install : qui crée la machine, la lance et démarre l'installation automatique.

```
#/bin/bash
# fichier autovm.sh
## usage : autovm.sh [type] [nom domaine]
##         autovm.sh $1 $2
## Création et installation automatisée Fedora/Centos 7
## types (profils, baselines):
##         small, medium ou large
##
## Variables générales
## 1. $name : Nom du domaine
name=$2
## 2. $type : type d'installation
type=$1
## 3. $vol : Emplacement des disques
vol=/var/lib/libvirt/images
## 4. $conf : Emplacement HTTP des fichiers Kickstart
## Serveur Web sur l'hyperviseur (adresse du réseau "Default")
conf=https://192.168.122.1/conf
## 5. $mirror : Sources d'installation HTTP
mirror=https://192.168.122.1/repo
## Miroirs publics
#mirror=https://centos.mirrors.ovh.net/ftp.centos.org/7/os/x86_64
#mirror=https://ftp.belnet.be/ftp.centos.org/7/os/x86_64
#mirror=https://mirror.i3d.net/pub/centos/7/os/x86_64
## 6. $temp : nom temporaire pour le fichier Kickstart
temp="$name-$(uuidgen | cut -d - -f 1)"
## 7. $www : emplacement physique des fichiers de configuration
www=/var/www/html/conf

gest_dom ()
{
## Gestion des noms utilisés, ici pour un lab (à améliorer)
## Arrêt et retrait de la VM
echo "Arrêt et retrait de la VM $name"
/bin/virsh destroy $name; /bin/virsh undefine $name
#Erase the VM disk
rm -f $vol/$name.*
}


prep_ks ()
{
## Préparation du fichier Kickstart
echo "Préparation du fichier Kickstart"

##
touch $www/$temp.ks
cat << EOF > $www/$temp.ks
install
keyboard --vckeymap=be-oss --xlayouts='be (oss)'
reboot
rootpw --plaintext testtest
timezone Europe/Brussels
```

```
url --url="$mirror"
lang fr_BE
firewall --disabled
network --bootproto=dhcp --device=eth0
network --hostname=$name
# network --device=eth0 --bootproto=static --ip=192.168.22.10 --netmask 255.255.255.0 --gat\
eway 192.168.22.254 --nameserver=192.168.22.11 --ipv6 auto
auth  --useshadow  --passalgo=sha512
text
firstboot --enable
skipx
ignoredisk --only-use=vda
bootloader --location=mbr --boot-drive=vda
zerombr
clearpart --all --initlabel
part /boot --fstype="xfs" --ondisk=vda --size=500
part swap --recommended
part pv.00 --fstype="lvmpv" --ondisk=vda --size=500 --grow
volgroup local0 --pesize=4096 pv.00
logvol /  --fstype="xfs"  --size=4000 --name=root --vgname=local0
%packages
@core
%end
%post
#yum -y update
mkdir /root/.ssh
curl $conf/id_rsa.pub > /root/.ssh/authorized_keys
sed -i 's/PasswordAuthentication yes/PasswordAuthentication no/g' /etc/ssh/sshd_config
%end
EOF

chown apache:apache $www/$temp.ks
}

virt_install ()
{

installation ()
{
## Démarrage de l'installation du domaine
echo "Démarrage de l'installation du domaine $name"
## Installation et lancement silencieux en mode texte
## selon le profil (baseline) défini dans la variable $type
nohup \
/bin/virt-install \
--virt-type kvm \
--name=$name \
--disk path=$vol/$name.$format,size=$size,format=$format \
--ram=$ram \
--vcpus=$vcpus \
--os-variant=rhel7 \
--network bridge=$bridge \
--graphics none \
```

```
--noreboot \
--console pty,target_type=serial \
--location $mirror \
-x "ks=$conf/$temp.ks console=ttyS0,115200n8 serial" \
> /dev/null 2>&1 &

## choix installation cdrom avec Kickstart local
#ks=/var/www/html/conf
#iso=path/to/iso
#--cdrom $iso \
#--initrd-inject=/$temp.ks -x "ks=file:/$temp.ks console=ttyS0,115200n8 serial" \
}

if [ $type = small ] ; then
        size=8
        format=qcow2
        ram=1024
        vcpus=1
        bridge=virbr0
    installation
elif [ $type = medium ] ; then
        size=16
        format=qcow2
        ram=2048
        vcpus=2
        bridge=virbr0
    installation
elif [ $type = large ] ; then
        size=32
        format=qcow2
        ram=4096
        vcpus=4
        bridge=virbr0
    installation
else
        exit
fi
}

gest_dom
prep_ks
virt_install

# rm -f $www/$temp.ks
```

15. Surveillance

```
virt-top
```

16. Commandes Virsh

```
virsh# version
```

```
Compiled against library: libvirt 2.0.0
Using library: libvirt 2.0.0
Utilisation de l'API : QEMU 2.0.0
Exécution de l'hyperviseur : QEMU 1.5.3
```

16.1. Domain Management (help keyword 'domain')

- `attach-device` attacher un périphérique depuis un fichier XML
- `attach-disk` attacher un périphérique disque
- `attach-interface` attacher une interface réseau
- `autostart` démarrer automatiquement un domaine
- `blkdeviotune` Set or query a block device I/O tuning parameters.
- `blkiotune` Get or set blkio parameters
- `blockcommit` Start a block commit operation.
- `blockcopy` Start a block copy operation.
- `blockjob` Manage active block operations
- `blockpull` Populate a disk from its backing image.
- `blockresize` Modifie la taille d'un périphérique bloc de domaine.
- `change-media` Change media of CD or floppy drive
- `console` se connecter à la console invitée
- `cpu-baseline` compute baseline CPU
- `cpu-compare` compare host CPU with a CPU described by an XML file
- `cpu-stats` show domain cpu statistics
- `create` créer un domaine depuis un fichier XML
- `define` définir (mais ne pas démarrer) un domaine depuis un fichier XML
- `desc` show or set domain's description or title
- `destroy` destroy (stop) a domain
- `detach-device` détacher un périphérique depuis un fichier XML
- `detach-disk` détacher un périphérique disque
- `detach-interface` détacher une interface réseau
- `domdisplay` domain display connection URI
- `domfsfreeze` Freeze domain's mounted filesystems.
- `domfsthaw` Thaw domain's mounted filesystems.
- `domfsinfo` Get information of domain's mounted filesystems.
- `domfstrim` Invoke fstrim on domain's mounted filesystems.
- `domhostname` print the domain's hostname
- `domid` convertir un nom de domaine ou UUID en ID de domaine
- `domif-setlink` set link state of a virtual interface
- `domiftune` get/set parameters of a virtual interface
- `domjobabort` abort active domain job
- `domjobinfo` domain job information
- `domname` convertir l'ID ou l'UUID du domaine en nom de domaine
- `domrename` rename a domain
- `dompmsuspend` suspend a domain gracefully using power management functions
- `dompmwakeup` wakeup a domain from pmsuspended state
- `domuuid` convertir un ID ou un nom de domaine en UUID de domaine

- `domxml-from-native` Convert native config to domain XML
- `domxml-to-native` Convert domain XML to native config
- `dump` vider l'espace mémoire d'un domaine dans un fichier pour analyse
- `dumpxml` informations du domaine en XML
- `edit` edit XML configuration for a domain
- `event` Domain Events
- `inject-nmi` Inject NMI to the guest
- `iothreadinfo` view domain IOThreads
- `iothreadpin` control domain IOThread affinity
- `iothreadadd` add an IOThread to the guest domain
- `iothreaddel` delete an IOThread from the guest domain
- `send-key` Send keycodes to the guest
- `send-process-signal` Send signals to processes
- `lxc-enter-namespace` LXC Guest Enter Namespace
- `managedsave` managed save of a domain state
- `managedsave-remove` Remove managed save of a domain
- `memtune` Get or set memory parameters
- `perf` Get or set perf event
- `metadata` show or set domain's custom XML metadata
- `migrate` migrer un domaine vers un autre hôte
- `migrate-setmaxdowntime` set maximum tolerable downtime
- `migrate-compcache` get/set compression cache size
- `migrate-setspeed` Set the maximum migration bandwidth
- `migrate-getspeed` Get the maximum migration bandwidth
- `migrate-postcopy` Switch running migration from pre-copy to post-copy
- `numatune` Get or set numa parameters
- `qemu-attach` QEMU Attach
- `qemu-monitor-command` QEMU Monitor Command
- `qemu-monitor-event` QEMU Monitor Events
- `qemu-agent-command` QEMU Guest Agent Command
- `reboot` redémarrer un domaine
- `reset` reset a domain
- `restore` restaurer un domaine à partir d'un état sauvé dans un fichier
- `resume` réactiver un domaine
- `save` enregistrer l'état du domaine dans un fichier
- `save-image-define` redefine the XML for a domain's saved state file
- `save-image-dumpxml` saved state domain information in XML
- `save-image-edit` edit XML for a domain's saved state file
- `schedinfo` montrer/définir les paramètres du planificateur
- `screenshot` take a screenshot of a current domain console and store it into a file
- `set-user-password` set the user password inside the domain
- `setmaxmem` changer la limite maximum de mémoire
- `setmem` changer la mémoire allouée
- `setvcpus` changer le nombre de processeurs virtuels
- `shutdown` arrêter un domaine proprement
- `start` démarrer un domaine (précédemment défini)
- `suspend` suspendre un domaine
- `ttyconsole` console TTY
- `undefine` undefine a domain
- `update-device` update device from an XML file

- `vcpucount` domain vcpu counts
- `vcpuinfo` detailed domain vcpu information
- `vcpupin` control or query domain vcpu affinity
- `emulatorpin` control or query domain emulator affinity
- `vncdisplay` affichage vnc
- `guestvcpus` query or modify state of vcpu in the guest (via agent)

16.2. Domain Monitoring (help keyword 'monitor')

- `domblkerror` Show errors on block devices
- `domblkinfo` domain block device size information
- `domblklist` list all domain blocks
- `domblkstat` retourner les statistiques d'un périphérique en mode bloc pour un domaine
- `domcontrol` domain control interface state
- `domif-getlink` get link state of a virtual interface
- `domifaddr` Get network interfaces' addresses for a running domain
- `domiflist` list all domain virtual interfaces
- `domifstat` obtenir les statistiques d'une interface réseau pour un domaine
- `dominfo` informations du domaine
- `dommemstat` get memory statistics for a domain
- `domstate` état du domaine
- `domstats` get statistics about one or multiple domains
- `domtime` domain time
- `list` lister les domaines

16.3. Host and Hypervisor (help keyword 'host')

- `allocpages` Manipulate pages pool size
- `capabilities` capacités
- `cpu-models` CPU models
- `domcapabilities` domain capabilities
- `freecell` Mémoire NUMA disponible
- `freepages` NUMA free pages
- `hostname` afficher le nom d'hôte de l'hyperviseur
- `maxvcpus` connection vcpu maximum
- `node-memory-tune` Get or set node memory parameters
- `nodecpumap` node cpu map
- `nodecpustats` Prints cpu stats of the node.
- `nodeinfo` informations du noeud
- `nodememstats` Prints memory stats of the node.
- `nodesuspend` suspend the host node for a given time duration
- `sysinfo` print the hypervisor sysinfo
- `uri` afficher l'URI canonique de l'hyperviseur
- `version` afficher la version

16.4. Interface (help keyword ‘interface’)

- `iface-begin` create a snapshot of current interfaces settings, which can be later committed (iface-commit) or restored (iface-rollback)
- `iface-bridge` create a bridge device and attach an existing network device to it
- `iface-commit` commit changes made since iface-begin and free restore point
- `iface-define` define an inactive persistent physical host interface or modify an existing persistent one from an XML file
- `iface-destroy` destroy a physical host interface (disable it / “if-down”)
- `iface-dumpxml` interface information in XML
- `iface-edit` edit XML configuration for a physical host interface
- `iface-list` list physical host interfaces
- `iface-mac` convert an interface name to interface MAC address
- `iface-name` convert an interface MAC address to interface name
- `iface-rollback` rollback to previous saved configuration created via iface-begin
- `iface-start` start a physical host interface (enable it / “if-up”)
- `iface-unbridge` undefine a bridge device after detaching its slave device
- `iface-undefine` undefine a physical host interface (remove it from configuration)

16.5. Network Filter (help keyword ‘filter’)

- `nwfilter-define` define or update a network filter from an XML file
- `nwfilter-dumpxml` network filter information in XML
- `nwfilter-edit` edit XML configuration for a network filter
- `nwfilter-list` list network filters
- `nwfilter-undefine` undefine a network filter

16.6. Networking (help keyword ‘network’)

- `net-autostart` démarrer automatiquement un réseau
- `net-create` créer un réseau depuis un fichier XML
- `net-define` define an inactive persistent virtual network or modify an existing persistent one from an XML file
- `net-destroy` destroy (stop) a network
- `net-dhcp-leases` print lease info for a given network
- `net-dumpxml` informations du réseau en XML
- `net-edit` edit XML configuration for a network
- `net-event` Network Events
- `net-info` network information
- `net-list` lister les réseaux
- `net-name` convertir l’UUID d’un réseau en nom de réseau
- `net-start` démarrer un réseau inactif (précédemment défini)
- `net-undefine` undefine a persistent network
- `net-update` update parts of an existing network’s configuration
- `net-uuid` convertir le nom d’un réseau en UUID de réseau

16.7. Node Device (help keyword ‘nodedev’)

- `nodedev-create` create a device defined by an XML file on the node
- `nodedev-destroy` destroy (stop) a device on the node
- `nodedev-detach` detach node device from its device driver
- `nodedev-dumpxml` node device details in XML
- `nodedev-list` enumerate devices on this host
- `nodedev-reattach` reattach node device to its device driver
- `nodedev-reset` reset node device

16.8. Secret (help keyword ‘secret’)

- `secret-define` define or modify a secret from an XML file
- `secret-dumpxml` secret attributes in XML
- `secret-get-value` Output a secret value
- `secret-list` list secrets
- `secret-set-value` set a secret value
- `secret-undefine` undefine a secret

16.9. Snapshot (help keyword ‘snapshot’)

- `snapshot-create` Create a snapshot from XML
- `snapshot-create-as` Create a snapshot from a set of args
- `snapshot-current` Get or set the current snapshot
- `snapshot-delete` Delete a domain snapshot
- `snapshot-dumpxml` Dump XML for a domain snapshot
- `snapshot-edit` edit XML for a snapshot
- `snapshot-info` snapshot information
- `snapshot-list` List snapshots for a domain
- `snapshot-parent` Get the name of the parent of a snapshot
- `snapshot-revert` Revert a domain to a snapshot

16.10. Storage Pool (help keyword ‘pool’)

- `find-storage-pool-sources-as` find potential storage pool sources
- `find-storage-pool-sources` discover potential storage pool sources
- `pool-autostart` démarrer automatiquement un pool
- `pool-build` construire un pool
- `pool-create-as` créer un pool depuis un ensemble d’arguments
- `pool-create` créer un pool depuis un fichier XML
- `pool-define-as` définir un pool à partir d’un ensemble d’argument
- `pool-define` define an inactive persistent storage pool or modify an existing persistent one from an XML file
- `pool-delete` effacer un pool
- `pool-destroy` destroy (stop) a pool
- `pool-dumpxml` informations du pool en XML
- `pool-edit` edit XML configuration for a storage pool
- `pool-info` informations du pool de stockage
- `pool-list` lister les pools

- `pool-name` convertir l'UUID d'un pool en nom de pool
- `pool-refresh` rafraichir un pool
- `pool-start` démarrer un pool inactif (précédemment défini)
- `pool-undefine` supprimer un pool inactif
- `pool-uuid` convertir le nom d'un pool en UUID de pool
- `pool-event` Storage Pool Events

16.11. Storage Volume (help keyword 'volume')

- `vol-clone` cloner un volume.
- `vol-create-as` créer un volume depuis un ensemble d'arguments
- `vol-create` créer un volume depuis un fichier XML
- `vol-create-from` create a vol, using another volume as input
- `vol-delete` supprimer un volume
- `vol-download` download volume contents to a file
- `vol-dumpxml` informations du volume en XML
- `vol-info` informations du volume de stockage
- `vol-key` renvoie la clé du volume pour un nom ou un chemin donné de volume
- `vol-list` lister les volumes
- `vol-name` returns the volume name for a given volume key or path
- `vol-path` renvoie le chemin vers le volume pour un nom ou une clé donnée de volume
- `vol-pool` renvoie le pool de stockage pour un nom ou un chemin donné de volume
- `vol-resize` modifier la taille d'un volume
- `vol-upload` upload file contents to a volume
- `vol-wipe` wipe a vol

16.12. Virsh itself (help keyword 'virsh')

- `cd` change the current directory
- `echo` echo arguments
- `exit` quitter ce terminal interactif
- `help` imprimer l'aide
- `pwd` print the current directory
- `quit` quitter ce terminal interactif
- `connect` (re)connecter à l'hyperviseur

Cinquième partie Disques et stockage LVM

Objectifs de certification

Linux Essentials

- *Sujet 4 : Le système d'exploitation Linux*
  - 4.3 Localisation des données (valeur : 3)

RHCSA EX200

- **4. Configurer le stockage local**
  - 4.1. Lister, créer, supprimer des partitions sur des disques MBR et GPT
  - 4.2. Créer et supprimer des volumes physiques
  - 4.3. Attribuer des volumes physiques aux groupes de volumes
  - 4.4. Créer et supprimer des volumes logiques
  - 4.5. Configurer des systèmes pour monter des systèmes de fichiers au démarrage par identificateur UUID ou étiquette
  - 4.6. Ajouter de nouvelles partitions, de nouveaux volumes logiques et de la swap à un système de manière non destructive
- **5. Créer et configurer des systèmes de fichiers**
  - 5.1. Créer, monter, démonter et utiliser des systèmes de fichiers `vfat`, `ext4` et `xfs`
  - 5.2. Monter et démonter des systèmes de fichiers réseau NFS
  - 5.3. Étendre des volumes logiques existants
  - 5.4. Créer et configurer des répertoires Set-GID pour la collaboration
  - 5.5. Configurer la compression de disque
  - 5.7. Gérer du stockage en couche (layered storage)

LPIC 1

- *Sujet 104 : Disques, systèmes de fichiers Linux , arborescence de fichiers standard (FHS)*
  - 104.1 Création des partitions et des systèmes de fichiers
  - 104.2 Maintenance de l'intégrité des systèmes de fichiers
  - 104.3 Montage et démontage des systèmes de fichiers

LPIC 2

- *Sujet 203 : Systèmes de fichiers et périphériques*
  - 203.1 Intervention sur le système de fichiers Linux (valeur : 4)
  - 203.2 Maintenance des systèmes de fichiers Linux (valeur : 3)
  - 203.3 Options de création et de configuration des systèmes de fichiers (valeur : 2)
- *Sujet 204 : Administration avancée des périphériques de stockage*
  - 204.1 Configuration du RAID logiciel (valeur : 3)
  - 204.2 Ajustement des accès aux périphériques de stockage (valeur : 2)
  - 204.3 Gestionnaire de volumes logiques (valeur : 3)

Introduction

Cette partie sur les disques et le stockage LVM porte sur les différents types de systèmes de fichiers, le formatage, le schéma de partitionnement, les tables MBR et GPT, les points de montage, la mémoire SWAP, la manipulation des disques RAID 1, RAID 5 et RAID 10. On trouvera aussi un développement sur le stockage LVM sur un plan théorique et pratique.

5. Disques sous Linux

On trouvera ici l'essentiel des concepts et des outils pour manipuler des disques sous Linux : périphériques blocs, partitions EXT et XFS, mémoire swap, points de montages, quotas.

1. Rappels théoriques

1.1. Commandes à retenir

Commande	Description
`cat /proc/partitions`	Affiche la liste des disques et partitions avec leur id
`ls /dev/{v,s}d*`	Affiche les partitions sous forme de fichier de type bloc
`blkid`	Affiche les identifiants uniques (UUID) des partions
`lsblk`	Affiche la list des partitions et disques
`findmnt`	Affiche les points de montage
`df -h`	Affiche les points de montage avec des informations sur le système de fichier
`du`	Affiche l'espace réel occupé par des fichiers
`fdisk`	Permet de manipuler tables de partitions MBR
`gdisk`	Permet de manipuler tables de partitions GPT
`mkfs.*`	Binaires qui permettent de préparer des systèmes de fichier
`cat /etc/fstab`	Affiche les points de montage autmatiques au démarrage
`cat /etc/mtab`	Affiche les points de montage courant du système
`mount`	Commande qui permet de monter des systèmes de fichier
`mount -a`	Monte les systèmes de fichier renseigné dans `/etc/fstab`
`mkswap`	Fabrique une mémoire swapd
`swapon`	Monte une mémoire swap
`swapoff`	Démonte une mémoire swap
`partprobe`	Programme qui la la table de partition et en informe le noyau
`fsck.*`	Vérifie un système de fichier EX
`dumpe2fs`	Affiche les informations sur les blocs et super-blocs EXT
`xfsdump, xfsrestore`	Permet de sauvegarder/restaurer un système de fichier XFS
`debugfs`	Diagnostic sur des systèmes de fichier EXT
`tune2fs`	Configure des paramètres de configuration des systèmes de fichier EXT
`xfs_info`, `xfs_check` et `xfs_repair`	Diagnostic de systèmes de fichiers EXT
`smartd`, `smartctl`	Utilitaires pour gérer "SMART self-test and error logs"

Commande	Description
`mdadm.conf`	Fichier de configration RAID logiciel
`mdadm`	Permet de manipuler les configuration RAID logiciel
`cat /proc/mdstat`	Affiche les informations RAID logiciel

1.2. Concepts

- Partitions
- Systèmes de fichier EXT3, EXT4, BTRFS, XFS, NFS
- Structure du système de fichier
- Le système de fichier EXT4
- Le système de fichier XFS
- Le système de fichier NFS
- Le système de fichier CIFS
- LVM
- Opérations LVM
- RAID
- Swap
- Quotas

1.3. Partion

Un disque est composé d'une ou plusieurs partitions dont la table est contenue dans le MBR (Master Boot Record) ou dans le GUID Partition Table (GPT) :

- Le MBR (Master Boot Record) définit des tables primaires, étendue, logiques.
- On utilse `fdisk` pour configurer le MBR et `gdisk` pour configurer le GPT.

Les caractéristiques d'une partition sont :

- La taille en secteurs
- Un drapeau qui indique si elle est active
- Le type de partition

Une partition peut être utilisée pour héberger :

- un système de fichiers
- un espace Swap
- une application

1.4. Systèmes de fichiers

Les données sont normalement présentées à l'utilisateur et aux programmes selon une organisation structurée, sous la forme de répertoires et de fichiers. Pour pouvoir stocker ces données structurées sur un périphérique, il faut utiliser un format qui les représente sous la forme d'une succession de blocs de données : c'est ce qu'on appelle un système de fichiers (FS). Un FS est concrètement une arborescence de fichiers stockée typiquement dans une partition ou un volume logique (LV).

Est associé à chaque système de fichiers :

- un pilote du noyau
- des structures de données mémoire et disque
- des utilitaires qui permettent la création et la maintenance du FS, voire sa sauvegarde

“Formater” un FS, c’est formater une partition en écrivant sur le disque les tables système (Superbloc, table d’inode, répertoire racine, …) associé à son type.

Un FS contient différentes tables système :

- Le super-bloc qui contient les données générales (taille, montage, …)
- La table des inodes qui contient la table de description et d’allocation des fichiers, chaque fichier étant représenté par un numéro d’inoeuds (inode).
- Un répertoire est une table de correspondance de fichiers/numéro d’inoeud.

Les fichiers hébergés par un FS ne sont accessibles que s’ils sont montés c’est-à-dire s’il est associé à un répertoire (répertoire de montage).

1.5. Types de FS

- Ext2
- Journalisés :
  - ext3
  - ext4
  - reiserfs
  - xfs
  - btrfs
- Microsoft :
  - msdos
  - fat
  - ntfs
- CD-Rom :
  - iso9660
- Réseau :
  - nfs
  - cifs
- Cluster :
  - GlusterFS
- Système :
  - proc
  - sys
  - udev
  - selinux
  - cgroup
  - cpuset
- Spéciaux :
  - tmpfs
  - unionfs (persistence live-usb)
  - aufs
  - cachefs
  - cramfs
  - squashfs
  - fuse
- Loop

1.6. FS à journalisation

ext3 est une évolution de **ext2** et a pour principale différence l'utilisation d'un fichier journal, lui permettant ainsi d'éviter la longue phase de récupération lors d'un arrêt brutal de la machine.

Bien que ses performances soient moins appréciées que celles de certains de ses compétiteurs, comme **ReiserFS** ou **XFS**, il a l'avantage majeur de pouvoir être utilisé à partir d'une partition ext2, sans avoir à sauvegarder et à restaurer des données (un système de fichiers ext3 peut être monté et utilisé comme un système de fichiers ext2). Tous les utilitaires de maintenance pour les systèmes de fichiers ext2, comme `fsck`, peuvent également être utilisés avec ext3.

ext3 alloue les blocs libres juste à côté des autres blocs utilisés par le fichier, ce qui a pour effet de minimiser l'espace physique entre les blocs.

Beaucoup moins assujetti, il est néanmoins par définition fragmenté, c'est pourquoi son successeur **ext4** inclut un utilitaire de défragmentation natif travaillant au niveau des bits et gérant la défragmentation à chaud.

ext4 garde une compatibilité avec son prédécesseur et est considéré par ses propres concepteurs comme une étape intermédiaire devant mener à un vrai système de fichiers de nouvelle génération tel que **Btrfs**. Toutefois, ext4 est une étape utile et non une simple solution temporaire.

1.7. Table de comparaison des FS

Inspiré de https ://doc.ubuntu-fr.org/systeme_de_fichiers#comparaison_de_systemes_de_fichiers

Nom du système de fichiers	Taille maximale d'un fichier	Taille maximale d'une partition	Journalisée ou non ?	Gestion des droits d'accès ?	Notes
ext2fs (Extended File System)	2 TiB	4 TiB	Non	Oui	Extended File System est le système de fichiers natif de Linux. En ses versions 1 et 2, on peut le considérer comme désuet, car il ne dispose pas de la journalisation. Ext2 peut tout de même s'avérer utile sur des disquettes 3½ et sur les autres périphériques dont l'espace de stockage est restreint, car aucun espace ne doit être réservé à un journal, par de l'embarqué en temps réel.

Nom du système de fichiers	Taille maximale d'un fichier	Taille maximale d'une partition	Journalisée ou non ?	Gestion des droits d'accès ?	Notes
ext3fs	2 TiB	4 TiB	Oui	Oui	ext3 est essentiellement ext2 avec la gestion de la journalisation. Il est possible de passer une partition formatée en ext2 vers le système de fichiers ext3 (et vice versa) sans formatage.
ext4fs	16 TiB	1 EiB	Oui	Oui	ext4 est le successeur du système de fichiers ext3. Il est cependant considéré par ses propres concepteurs comme une solution intermédiaire en attendant le vrai système de nouvelle génération que sera Btrfs.

Nom du système de fichiers	Taille maximale d'un fichier	Taille maximale d'une partition	Journalisée ou non ?	Gestion des droits d'accès ?	Notes
ReiserFS	8 TiB	16 TiB	Oui	Oui	Développé par Hans Reiser et la société Namesys, ReiserFS est reconnu particulièrement pour bien gérer les fichiers de moins de 4 ko. Un avantage du ReiserFS, par rapport à ext3, est qu'il ne nécessite pas une hiérarchisation aussi poussée : il s'avère intéressant pour le stockage de plusieurs fichiers temporaires provenant d'Internet. Par contre, ReiserFS n'est pas recommandé pour les ordinateurs portables, car le disque dur tourne en permanence, ce qui consomme beaucoup d'énergie.
XFS	8 EiB	16 EiB	oui	oui	Performant et flexible. Attention, il n'est pas possible de réduire une partition xfs

Nom du système de fichiers	Taille maximale d'un fichier	Taille maximale d'une partition	Journalisée ou non ?	Gestion des droits d'accès ?	Notes
FAT (File Allocation Table)	2 GiB	2 GiB	Non	Non	Développé par Microsoft, ce système de fichiers se rencontre moins fréquemment aujourd'hui. Il reste néanmoins utilisé sur les disquettes 3½ formatées sous Windows et devrait être utilisé sous Linux si une disquette doit aussi être lue sous Windows. Il est aussi utilisé par plusieurs constructeurs comme système de fichiers pour cartes mémoires (memory sticks), car, bien documenté, ce système de fichiers reste le plus universellement utilisé et accessible.

Nom du système de fichiers	Taille maximale d'un fichier	Taille maximale d'une partition	Journalisée ou non ?	Gestion des droits d'accès ?	Notes
FAT32	4 GiB	8 TiB	Non	Non	Ce système de fichiers, aussi créé par Microsoft, est une évolution de son prédécesseur. Depuis ses versions 2000 SP4 et XP, Windows ne peut pas formater (ou bloque volontairement le formatage) une partition en FAT32 d'une taille supérieure à 32 Go. Cette limitation ne s'applique pas sous Linux, de même qu'avec des versions antérieures de Windows. Une partition FAT32 d'une taille supérieure à 32 Go déjà formatée pourra être lue par Windows, peu importe sa version.
NTFS (New Technology File System)	16 TiB	256 TiB	Oui	Oui	Ce système de fichiers a aussi été développé par Microsoft, et il reste très peu documenté. L'écriture depuis Linux sur ce système de fichiers est stable à l'aide du pilote ntfs-3g. Ce pilote est inclus de base dans Ubuntu, et disponible en paquets dans les dépôts pour les versions antérieures.

Nom du système de fichiers	Taille maximale d'un fichier	Taille maximale d'une partition	Journalisée ou non ?	Gestion des droits d'accès ?	Notes
exFAT	16 TiB	256 TiB	Oui	Oui	Ce système de fichiers a aussi été développé par Microsoft. L'écriture depuis Linux sur ce système de fichiers est stable à l'aide du pilote exfat-fuse.

Légende des unités :

- EiB = Exbioctets (1024 pébioctets)
- PiB = Pébioctet (1024 tébioctet)
- TiB = Tébioctet (1024 gibioctets)
- GiB = Gibioctet (1024 mibioctets)

2. Auditer les disques

2.1. Lister les périphériques bloc

Lister les périphériques bloc :

```
sudo lsblk
sudo blkid
sudo find /dev -type b
```

2.2. Lister les disques et les partitions

Lister les disques et les partitions :

```
cat /proc/partitions
sudo fdisk -l
```

2.3. Lister les FS disponibles

Lister les FS disponibles :

```
cat /proc/filesystems
```

2.4. Lister les points de montages

Points de montages :

```
cat /proc/mounts
findmnt --fstab-
```

2.5. Lister les points de montage automatiques

Points de montage automatiques :

```
cat /etc/fstab
cat /etc/mtab
df -Th
```

2.6. Commandes sur les fichiers

Commandes sur les fichiers :

```
du -sh /nomdudossier
stat nomdefichier
```

3. Formatage Ext3/Ext4

La commande `mk2fs` fait appel à des programmes de plus bas niveau comme `mkfs.ext3` ou `mkfs.ext4`

Pour formater un périphérique en, on retiendra :

3.1. EXT2

```
mk2fs /dev/sdx1
```

3.2. en EXT3

```
mk2fs -j /dev/sdx1
```

ou

```
mkfs.ext3 /dev/sdx1
```

ou encore

```
mk2fs -t ext3 /dev/sdx1
```

3.3. en EXT4

```
mkfs.ext4 /dev/sdx1
```

3.4. Commandes espace-utilisateur du sytème de fichiers “EXT”

On retiendra d’autres commandes espace-utilisateur du sytème de fichiers “EXT” comme par exemple :

- `tune2fs -l /dev/sdx1` qui affiche les paramètres d’un FS ext3
- `e2fsck /dev/sdx1` qui vérifie ou répare `-y` un FS
- `dumpe2fs /dev/sdx1` qui affiche des informations sur un FS
- `e2label /dev/sdx1` qui affiche ou modifie l’étiquette d’un FS
- `tune2fs -c mmc` qui modifie les paramètres d’un FS (vérification après un nombre maximal de montage
- `resize2fs` redimensionne un FS
- `debugfs`
- `e2image` sauvegarde les métadonnées dans un fichier, lisible avec `debugfs -i` ou `dumpe2fs -i`
- `e2freefrag` affiche la fragmentation de la place libre
- `e2undo` rejoue le journal qui n’a pas été accompli
- `tune2fs -j /dev/sdx1` convertit un FS ext2 en FS ext3
- `tune2fs -O extents,uninit_bg,dir_index /dev/sdx1` convertit un FS ext3 en FS ext4

4. Formatage XFS

En Debian8 :

```
apt-get install xfsprogs
```

- `xfs_admin`
- `xfs_bmap`
- `xfs_copy`
- `xfs_db`
- `xfs_estimate`
- `xfs_freeze`
- `xfs_fsr`
- `xfs_growfs`
- `xfs_info`
- `xfs_io`
- `xfs_logprint`
- `xfs_metadump`
- `xfs_mdrestore`
- `xfs_mkfile`
- `xfs_ncheck`
- `xfs_quota`

5. Manipulation de périphériques Bloc

5.1. Copie par blocs avec dd

- `dd` est une commande unix permettant de copier un fichier (avec ou sans conversion au passage) notamment sur des périphériques blocs tel que des disques durs ou des lecteurs CD-ROM ou inversement.
- Contrairement à `cp`, la commande `dd` copie des portions de données brutes d’un périphérique. Par conséquent, `dd` préserve le système de fichier sous-jacent. `cp` se contente de traiter des données et les transfère d’un système de fichier à un autre.

5.2. Syntaxe de la commande `dd`

La syntaxe de `dd` est différente des autres commandes unix traditionnelles. `dd` utilise des options de la forme `option=valeur` au lieu des habituelles `-o` valeur ou `--option=valeur`.

Les principales options de dd sont les suivantes :

- `if=fichier_entree` (Input File) : lit ce fichier en entrée. Cela peut être un fichier régulier comme un périphérique de type bloc. Par défaut, c'est l'entrée standard qui est utilisée (par exemple le clavier).
- `of=fichier_sortie` (Output File) : écrit dans ce fichier en sortie.
- `bs=t_b` (Block Size) : copie les données par blocs de `t_b` octets.
- `count=n_b` : ne copie que `n_b` blocs.
- `skip=n_e` : ignore les `n_e` premiers blocs du fichier d'entrée1 (Ne copie le fichier d'entrée qu'à partir du bloc de rang `n_e+1`.)
- `seek=n_s` : ignore les `n_s` premiers blocs du fichier de sortie1 (Ne commence à écrire dans le fichier de sortie qu'à partir du bloc de rang `n_s+1`.)

5.3. Créer un fichier rempli de bits aléatoires

Créer un fichier rempli de bits aléatoires de 1 Mo :

```
dd if=/dev/urandom bs=1024 count=1000 of=fichier.bin
```

5.4. Créer une clé bootable

Créer une clé bootable Centos 7 sur /dev/sdb :

```
sudo dd if=CentOS-7.0-1406-x86_64-DVD.iso of=/dev/sdb
```

5.5. Créer une SD Card

Créer une SD Card Raspberry Pi :

```
sudo dd bs=4M if=~/2012-12-16-wheezy-raspbian.img of=/dev/sdb
```

5.6. Créer l'image d'un disque (démonté)

Créer l'image d'un disque (démonté) :

```
sudo dd if=/dev/sdb of=~/sdb.img
```

5.7. Copier un disque sur l'autre

Copier un disque sur l'autre :

```
sudo dd if=/dev/sdb of=/dev/sdc conv=noerror,sync
```

5.8. Copier une partition

Copier une partition :

```
sudo dd if=/dev/sdb1 of=~/sdb1.img
```

5.9. Créer des fichiers-disques vides d'une taille arbitraire

L'option `seek` permet de comencer à écrire sur la cible à partir du bloc obtenu par la multiplication de la valeur `seek` et de la taille de bloc. Quand le nombre `count` est nul, alors rien n'est écrit. Cela permet de créer des fichiers-disques vides d'une taille arbitraire. Par exemple, avec une valeur `bs=1M`, `seek=1024` et `count=0`, on obtient un fichier de `1M*1024` octets soit 1Go. Le fichier sera totalement vide.

```
size=1
seek=$[${size}*1024]
disk="./disk1"
dd if=/dev/zero of=${disk} bs=1M seek=${seek} count=0


ls -lh disk1 ; du -sh disk1


-rw-rw-r--. 1 francois francois 1,0G  6 jun 22:12 disk1
0       disk1
```

6. Montage du système de fichier

6.1. Montage manuel du système de fichier

La commande `mount` permet de monter le FS d'un périphérique sous un répertoire local vide. C'est à partir de cet emplacement que le FS sera accessible.

La syntaxe de la commande `mount` est la suivante :

```
sudo mount -t <type> -o <option,option> /dev/<sdx1> /<repertoire_vide>
```

Les options habituelles, parmi d'autres, peuvent être `ro`, `rw`, `sync` (écriture synchrones sans passer par une mémoire cache) ou encore `loop` pour monter un fichier plutôt q'un périphérique bloc.

La commande `umount` démonte le périphérique désigné à condition qu'il ne soit plus utilisé.

6.2. Montage du système de fichier au démarrage

Au démarrage, les différents FS d'un système seront montés selon indications du fichier `/etc/fstab`. Ce fichier contient six champs :

```
# <périphérique> <point de montage> <type> <options> <dump> <fsck>
/dev/sda1 / xfs defaults 0 1
/dev/sda2 /opt xfs defaults 0 0
```

Si les quatre premiers champs obligatoires sont assez évident, on notera les deux derniers champs optionnels :

- **dump** active la sauvegarde
- **fsck** réalise la vérification automatique du FS au démarrage. En EXT, la valeur est toujours **1** pour le répertoire racine `/`

La commande `mount -a` va lire le fichier `/etc/fstab` et monter les FS indiqués.

Le fichier `/etc/mtab` contient tous les FS que le noyau utilise.

7. Créer un système de fichier loop

Création d'un fichier de 1Go :

```
dd if=/dev/zero of=${HOME}/fs_ext4.img bs=1M seek=1024 count=0
```

Formatage en ext4

```
sudo mkfs.ext4 ${HOME}/fs_ext4.img
```

```
mke2fs 1.42.9 (28-Dec-2013)
/home/francois/fs_ext4.img n'est pas un périphérique spécial en mode bloc.
Procéder malgré tout ? (o,n) o
Rejet des blocs de périphérique : complété
Étiquette de système de fichiers=
Type de système d'exploitation : Linux
Taille de bloc=4096 (log=2)
Taille de fragment=4096 (log=2)
« Stride » = 0 blocs, « Stripe width » = 0 blocs
65536 i-noeuds, 262144 blocs
13107 blocs (5.00%) réservés pour le super utilisateur
Premier bloc de données=0
Nombre maximum de blocs du système de fichiers=268435456
8 groupes de blocs
32768 blocs par groupe, 32768 fragments par groupe
8192 i-noeuds par groupe
Superblocs de secours stockés sur les blocs :
        32768, 98304, 163840, 229376

Allocation des tables de groupe : complété
Écriture des tables d'i-noeuds : complété
Création du journal (8192 blocs) : complété
Écriture des superblocs et de l'information de comptabilité du système de
fichiers : complété
```

Montage

```
sudo mkdir /mnt/ext4
sudo mount -t ext4 -o loop ${HOME}/fs_ext4.img /mnt/ext4
sudo losetup -a
```

```
/dev/loop0: [fc00]:917833 (/home/francois/fs_ext4.img)
```

Vérification

```
df -Th
```

```
Filesystem     Type      Size  Used Avail Use% Mounted on
udev           devtmpfs  981M   12K  981M   1% /dev
tmpfs          tmpfs     199M  1,5M  197M   1% /run
/dev/dm-0      ext4       18G  3,8G   13G  23% /
none           tmpfs     4,0K     0  4,0K   0% /sys/fs/cgroup
none           tmpfs     5,0M     0  5,0M   0% /run/lock
none           tmpfs     992M  152K  992M   1% /run/shm
none           tmpfs     100M   68K  100M   1% /run/user
/dev/sda1      ext2      236M   40M  184M  18% /boot
/dev/loop0     ext4      976M  1,3M  908M   1% /mnt/ext4
```

```
sudo -i su -c "echo $(date) > /mnt/ext4/f1"
sudo -i su -c "echo $(date) > /mnt/ext4/f2"
sudo mkdir /mnt/ext4/dir
sudo ls /mnt/ext4
```

```
dir  f1  f2  lost+found
```

Démontage et vérification du FS :

```
sudo umount ${HOME}/fs_ext4.img
sudo fsck ${HOME}/fs_ext4.img
```

```
fsck de util-linux 2.23.2
e2fsck 1.42.9 (28-Dec-2013)
/home/francois/fs_ext4.img : propre, 14/65536 fichiers, 12958/262144 blocs
```

Rendre le point de montage automatique au démarrage :

```
sudo -i su -c "echo ${HOME}/fs_ext4.img /mnt/ext4 ext4 rw 0 0 >> /etc/fstab"
mount -a
df -Th
```

8. Montage automatique du système de fichier

Le montage automatique (*autofs*) permet de définir des emplacements du système qui serviront de point de montage de manière opportune. Par exemple, lorsque l'emplacement `/data/backup` sera accédé, un périphérique bloc `/dev/sdx1` y sera monté.

Une table principale `/etc/auto.master` contient les emplacements et la référence pour cet emplacement dans une table secondaire qui contient les directives d'automontage. Le démon `autofs` vérifie en permanence l'accès à ces dossiers.

Ce mécanisme est utile notamment pour "automonter" des disques distants, des répertoire `/home` itinérants, etc.

Eventuellement en Centos 7, il faudra l'installer via `yum -y install autofs` et vérifier le démarrage du service via `systemctl status autofs`

Par exemple en Debian8 :

```
sudo apt -y install autofs
```

```
Paramétrage de autofs (5.0.8-2+deb8u1) ...
Creating config file /etc/auto.master with new version
Creating config file /etc/auto.net with new version
Creating config file /etc/auto.misc with new version
Creating config file /etc/auto.smb with new version
Creating config file /etc/default/autofs with new version
update-rc.d: warning: start and stop actions are no longer supported; falling back to defau\
lts
Traitement des actions différées (« triggers ») pour systemd (215-17+deb8u6) ...
```

```
cat  /etc/auto.{master,misc}
```

```
#
# Sample auto.master file
# This is an automounter map and it has the following format
# key [ -mount-options-separated-by-comma ] location
# For details of the format look at autofs(5).
#
#/misc        /etc/auto.misc
#
# NOTE: mounts done from a hosts map will be mounted with the
#       "nosuid" and "nodev" options unless the "suid" and "dev"
#       options are explicitly given.
#
#/net        -hosts
#
# Include /etc/auto.master.d/*.autofs
#
+dir:/etc/auto.master.d
#
# Include central master map if it can be found using
# nsswitch sources.
```

```
#
# Note that if there are entries for /net or /misc (as
# above) in the included master map any keys that are the
# same will not be seen as the first read key seen takes
# precedence.
#
+auto.master
#
# This is an automounter map and it has the following format
# key [ -mount-options-separated-by-comma ] location
# Details may be found in the autofs(5) manpage

cd               -fstype=iso9660,ro,nosuid,nodev        :/dev/cdrom

# the following entries are samples to pique your imagination
#linux                -ro,soft,intr                ftp.example.org:/pub/linux
#boot                -fstype=ext2                :/dev/hda1
#floppy                -fstype=auto                :/dev/fd0
#floppy                -fstype=ext2                :/dev/fd0
#e2floppy        -fstype=ext2                :/dev/fd0
#jaz                -fstype=ext2                :/dev/sdc1
#removable        -fstype=ext2                :/dev/hdd
```

9. Quotas sur les FS en EXT

Les quotas sur les disques se gèrent différemment d'un FS à l'autre.

Depuis que XFS est le FS par défaut on utilise les utilitaires XFS intégrés : https ://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/7/html/Storage_Administration_Guide/xfsquota.html

- 1. En FS EXT4, il faut les utilitaires disponibles :

```
sudo yum -y install quota || sudo apt -y install quota quotatool
```

- 1. Il sera nécessaire aussi d'activer les options `usrquota` et `grpquota` dans `/etc/fstab` :

```
grep quota /etc/fstab
```

```
/root/fs_ext4.img /mnt/ext4 ext4 rw,usrquota,grpquota 0 0
```

- 1. On crée (`-c`) la base de donnée de quota pour le FS de manière verbeuse (`-v`) des utilisateurs (`-u`) et groupes (`-u`) :

```
sudo quotacheck -cugv /mnt/ext4
```

- 1. Vérification :

```
sudo quotacheck -avug
```

Voici les limites proposées :

- **soft (souple)** : avertira les utilisateurs d'un dépassemnt souple (en Ko).
- **hard** : Empêchera l'usage supplémentaire du disque en cas de dépassemnt.

On peut aussi limiter le nombre d'inodes.

Enfin, on appelle la période de grâce le délai pendant lequel une limite souple peut être dépassée pour devenir une limite dure.

- 1. Configurer un utilisateur

```
sudo edquota -u jack
```

- 1. Configurer un groupe :

```
sudo edquota -u devel
```

- 1. Vérifier les quotas :

```
sudo repquota -as
```

```
*** Rapport pour les quotas user sur le périphérique /dev/loop1
Période de sursis bloc : 7days ; période de sursis inode : 7days
                        Space limits                File limits
Utilisateur     utilisé souple stricte sursis utilisé souple stricte sursis
------------------------------------------------------------------------
root      --     20K      0K      0K              2     0     0
```

- 1. Régler la période de sursis :

```
sudo edquota -t
```

10. Mémoire SWAP

La mémoire swap est un espace de stockage visant à pallier à un manque de mémoire vive du système. La mémoire swap sert à étendre la mémoire utilisable par un système d'exploitation par un fichier d'échange ou une partition dédiée.

- `mkswap <dev>` est la commande qui permet de créer un espace swap
- `swapon <dev>` permet d'activer une swap
- `swapoff <dev>` permet de désactiver une swap

Les arguments possibles pour désigner l'espace de stockage swap :

- un fichier
- un périphérique type bloc, un disque, une partition
- un LABEL avec l'option `-L`
- un UUID avec l'option `-U`

La commande `swapon -s` permet de voir la configuration des mémoire SWAP.

Sixième partie Configuration du réseau

Objectifs de certification

Linux Essentials

- *Sujet 4 : Le système d'exploitation Linux*
  - 4.4 Intégration au réseau (valeur : 2)

RHCSA EX200

- 7. **Gestion de base du réseau**
  - 7.1. Configurer les adresses IPv4 et IPv6
  - 7.2. Configurer la résolution du nom d'hôte
  - 7.3. Configurer des services réseau afin qu'ils se lancent automatiquement au démarrage

RHCE EX300

- 1. **Configuration et gestion de systèmes**
  - 1.2. Configurer des adresses IPv6 et résoudre des problèmes de base liés au protocole IPv6

LPIC 1

- *Sujet 109 : Notions élémentaires sur les réseaux*
  - 109.1 Notions élémentaires sur les protocoles Internet
  - 109.2 Configuration réseau persistante
  - 109.3 Résolution de problèmes réseaux simples
  - 109.4 Configuration de la résolution de noms

LPIC 2

- *Sujet 205 : Configuration réseau*
  - 205.1 Configuration réseau de base (valeur : 3)
  - 205.2 Configuration réseau avancée (valeur : 4)
  - 205.3 Résolution des problèmes réseau (valeur : 4)

Introduction

Cette partie porte sur la configuration réseau. On trouvera en premier lieu une introduction à TCP/IP et ensuite une synthèse rapide des commandes de diagnostic à retenir. Les trois chapitres qui suivent étudient la gestion du réseau sous Linux avec Network Manager, avec la librairie Iproute2 et Netplan. Enfin, cette partie se termine par la revue de différents outils réseau sous Linux.

Documentation

- https ://access.redhat.com/documentation/fr-FR/Red_Hat_Enterprise_Linux/7/html/Networking_Guide/index.html

6. Introduction à TCP/IP

Ce chapitre est une brève introduction à la pile des protocoles TCP/IP.

1. Protocoles Internet

Un protocole de communication est un ensemble de règles qui rendent les communications possibles car les intervenants sont censés les respecter.

Les protocoles définissent un sorte de langage commun que les intervenants utilisent pour se trouver, se connecter l'un à l'autre et y transporter des informations.

Les protocoles peuvent définir :

- des paramètres physiques comme des modulations, de type de supports physiques, des connecteurs, …
- le comportement d'un certain type de matériel
- des commandes
- des machines à état
- des types de messages
- des en-têtes qui comportent des informations utiles au transport

Ceux-ci sont discutés et élaborés par des organismes de standardisation.

Les protocoles TCP/IP sont formalisé par l'IETF dans des documents publics qui prennent le nom de RFC ("requests for comments"). On désigne ces documents par de numéro de référence. Tous les RFCs ne sont pas nécessairement des standards … pour un peu plus de détails sur les RFCs.

Les protocoles LAN / WAN / PAN sont formalisés par l'IEEE (IEEE 802), par l'ITU, l'ANSI, …

On distinguera ces organismes de standardisation de consortium commerciaux comme la WiFi Alliance ou des organismes étatiques nationaux et internationaux de régulation comme le FCC, l'ETSI, l'IBPT, etc.

1.1. Objectif de TCP/IP

- **Communiquer**
  - à l'échelle du globe
  - de manière libérale (ouverte)
- **quel que soit**
  - le contenu
  - le support
  - les hôtes

Objectif de TCP/IP

1.2. L'Internet

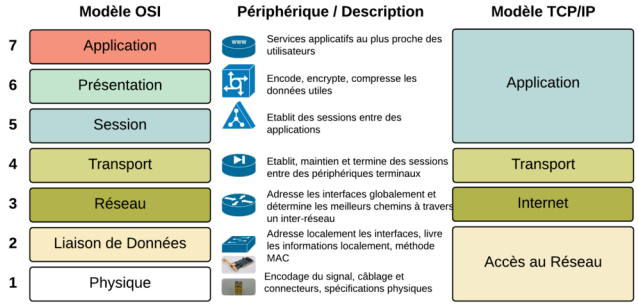
L'Internet est **l'interconnexion de réseaux à l'échelle du globe**. En IPv4, l'Internet a atteint sa taille limite.

1.3. Quatre couches

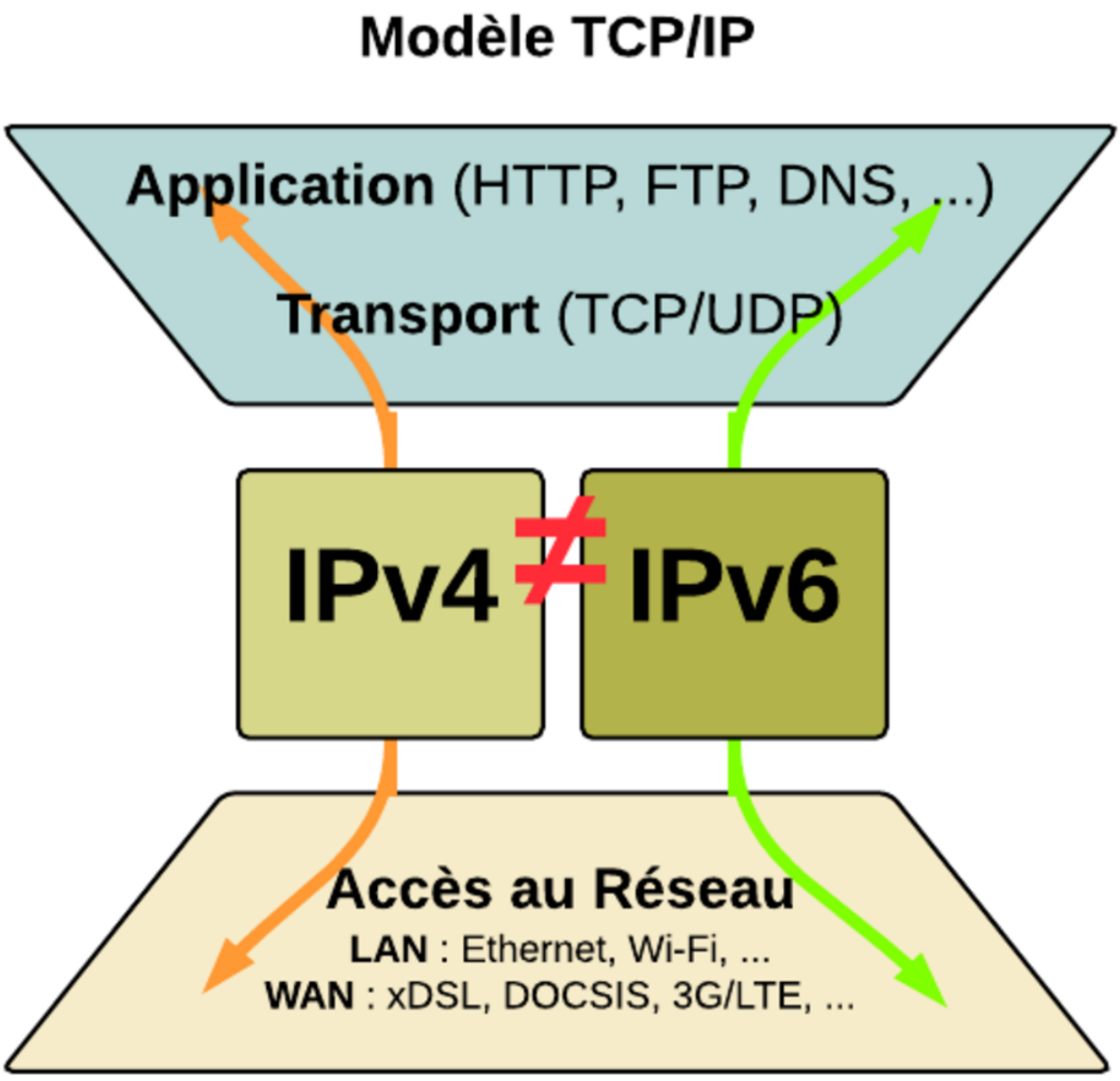
Le modèle de communication TCP/IP compte quatre couches.

- **Couche Application**
  - Elle est la couche de communication qui s'interface avec les utilisateurs.
  - Exemples de protocoles applicatifs : HTTP, DNS, DHCP, FTP, ...
  - S'exécute sur les machines hôtes.
- **Couche Transport : TCP**
  - Elle est responsable du dialogue entre les hôtes terminaux d'une communication.
  - Les applications utiliseront TCP pour un transport fiable et UDP sans ce service.
  - Les routeurs NAT et les pare-feu opèrent un filtrage au niveau de la couche transport.
- **Couche Internet : IP**
  - Elle permet de déterminer les meilleurs chemins à travers les réseau en fonction des adresses IPv4 ou IPv6 à portée globale.
  - Les routeurs transfèrent le trafic IP qui ne leur est pas destiné.
- **Couche Accès au réseau**
  - TCP/IP ne s'occupe pas de la couche Accès Réseau

- Elle organise le flux binaire et identifie physiquement les hôtes
- Elle place le flux binaire sur les support physique
- Les commutateurs, cartes réseau, connecteurs, câbles, etc. font partie de cette couche



Comparatif des modèles OSI et TCP/IP



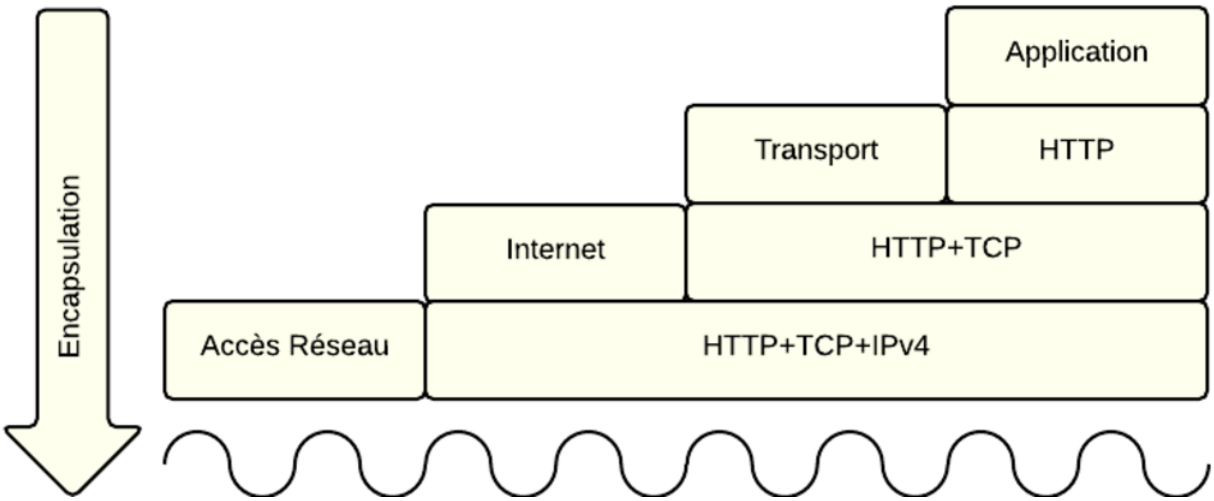
Modèle TCP/IP

Plus on monte dans les couches, plus on quitte les aspects matériels, plus on se rapproche de problématiques

logicielles.

1.4. Encapsulation

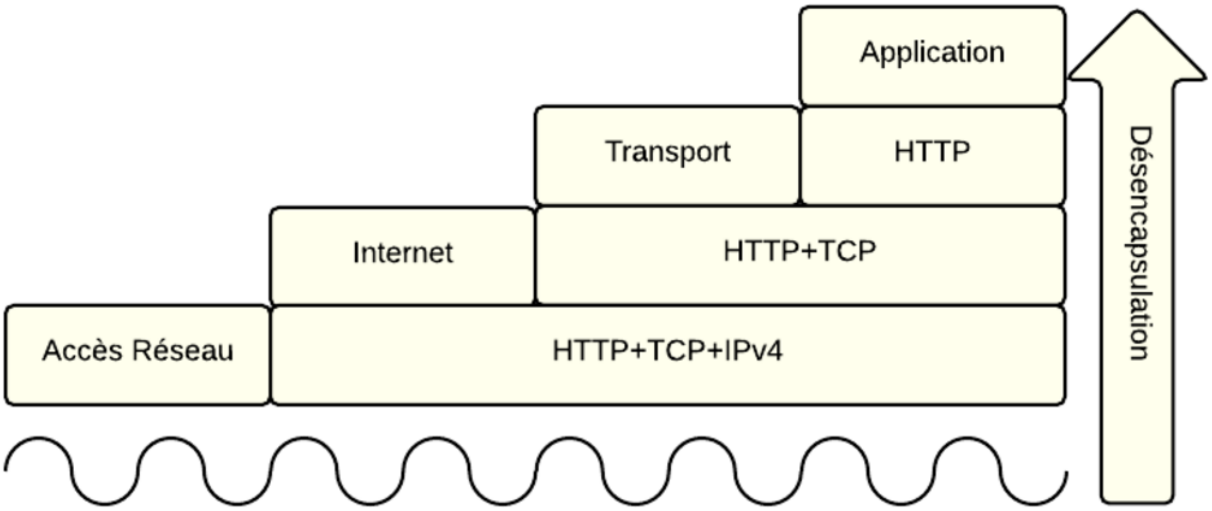
- Pour transmettre du contenu d'un ordinateur à un autre, l'utilisateur va utiliser un programme qui construit un message enveloppé par un en-tête applicatif, SMTP par exemple. Le message subit une première encapsulation.
- Le logiciel va utiliser un protocole de couche transport correspondant pour établir la communication avec l'hôte distant en ajoutant un en-tête TCP ou UDP.
- Ensuite, l'ordinateur va ajouter un en-tête de couche Internet, IPv4 ou IPv6 qui servira à la livraison des informations auprès de l'hôte destinataire. L'en-tête contient les adresses d'origine et de destination des hôtes.
- Enfin, ces informations seront encapsulées au niveau de la couche Accès qui s'occupera de livrer physiquement le message.



Encapsulation

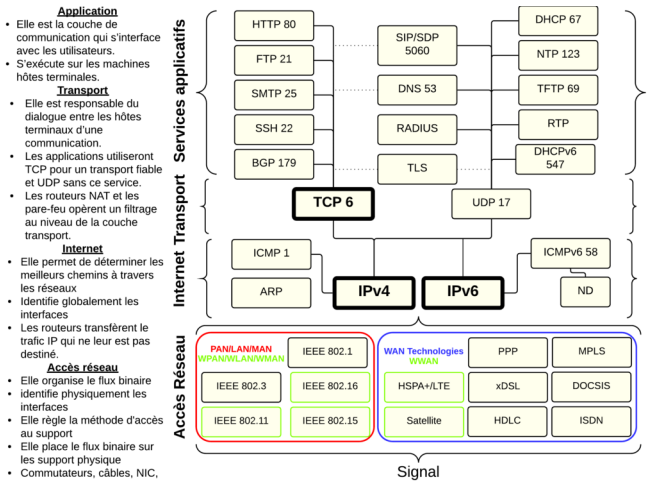
A la réception, l'hôte récepteur réalise l'opération inverse en vérifiant les en-têtes de chaque protocole correspondant à une des couches décrites. Ce processus s'appelle la désencapsulation.

Chaque couche ajoute une information fonctionnelle au message original. A la réception, l'hôte examine chaque couche et prend une décision quant à ce trafic.



Désencapsulation

Modèle TCP/IP détaillé



Modèle TCP/IP détaillé

2. Adressage et matériel

2.1. Adressage et identifiants

Les machines et leurs interfaces disposent d'identifiants au niveau de chaque couche :

- Couche Application : Nom de domaine, par exemple : **linux.goffinet.org**
- Couche Transport : Port TCP ou UDP, par exemple : **TCP80**
- Couche Internet : Adresse IPv4 et/ou IPv6, par exemple : **192.168.150.252/24** ou **2001:db8::1/64**
- Couche Accès : adresse physique (MAC), par exemple une adresse MAC 802 : **70:56:81:bf :7c :37**

2.2. Rôles des périphériques

IP voit deux rôles :

1. Les **hôtes terminaux** : nos ordinateurs au bout du réseau

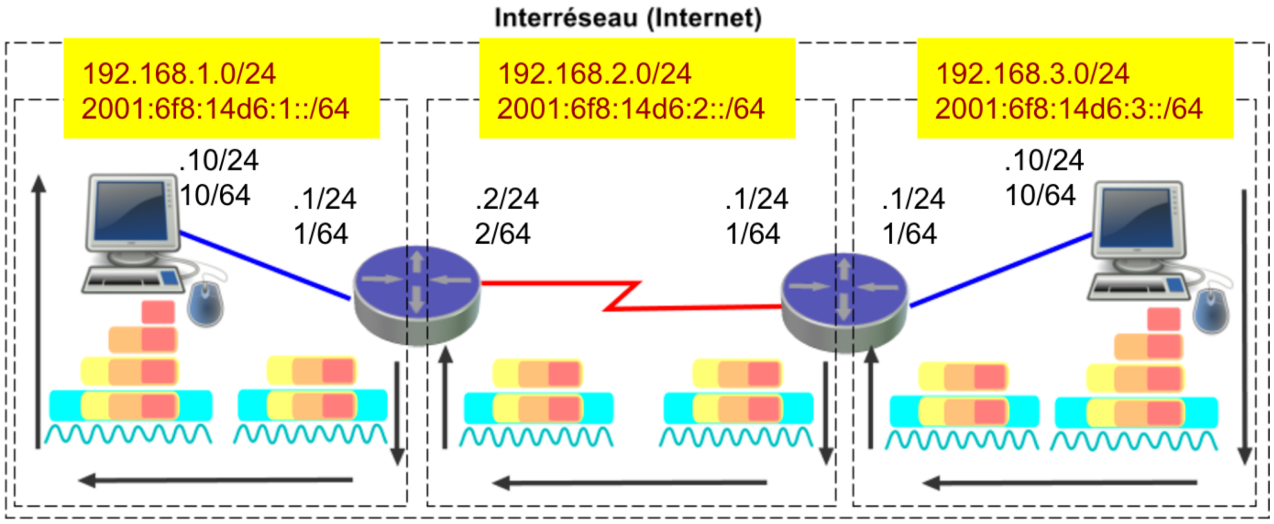
2. Les **routeurs chargés** de transférer les **paquets** en fonction de l'**adresse L3 IP (logique, hiérarchique)** de destination. Il permettent d'interconnecter les hôtes d'extrémité.

Aussi au sein du réseau local (LAN), le **commutateur** (switch) est chargé de transférer rapidement les **trames** Ethernet selon leur **adresse L2 MAC (physique)** de destination.

3. Routage IP

3.1. Domaines IP

- Deux noeuds (hôtes, interfaces, cartes réseau, PC, smartphone, etc.) doivent appartenir au même réseau, au même domaine IP pour communiquer directement entre eux.
- Quand les noeuds sont distants, ils ont besoin de livrer leur trafic à une passerelle, soit un routeur.
- D'une extrémité à l'autre, les adresses IP ne sont pas censées être modifiées (sauf NAT) par les routeurs. Par contre, le paquet est dés-encapsulé /ré-encapsulé différemment au niveau de la couche Accès au passage de chaque routeur.



Domaines IP et routage

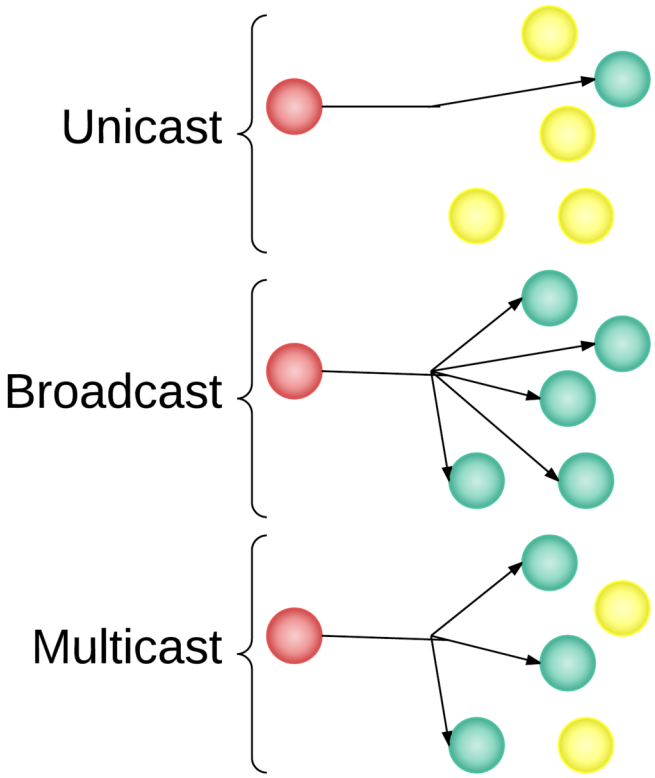
3.2. Type d'adresses IP

Les adresses IP permettent d'identifier de manière unique les hôtes d'origine et de destination. Les routeurs se chargent d'acheminer les paquets à travers les liaisons intermédiaires.

Il existe plusieurs types d'adresses qui correspondent à plusieurs usages.

On trouve au moins trois grandes catégories :

- les adresses *unicast* : à destination d'un seul hôte
- les adresses *broadcast* (IPv4) : à destination de tous les hôtes du réseau
- les adresses *multicast* (IPv4 et IPv6) : à destination de certains hôtes du réseau.



Parmi les adresses **unicast** on distinguera :

- les adresses *non-routées* : locales ou de loopback jamais transférées par les routeurs.
- les adresses *publiques* ou *globales* : pour lesquelles les routeurs publics acheminent les paquets
- les adresses *privées* ou *unique locale* : pour lesquelles seuls les routeurs privés transfèrent le trafic (les routeurs publics ne connaissent pas de chemin pour des destinations privées).

3.3. Nécessité du NAT en IPv4

A cause de la consommation galopante d'adresses IPv4 publiques, les autorités de l'Internet on décidé de proposer des solutions :

- L'adoption d'un **protocole nouveau IPv6** corrigeant les problèmes d'IPv4 avec un conseil de transition en double pile IPv4/IPV6.
- Entre autres solution d'offrir une connectivité IPv4 avec une seule adresse publique qui cache un réseau adressé avec des blocs privés. En vue d'offrir une connectivité globale (publique) à des hôtes privés, une des solutions est la traduction du trafic (NAT). Une autre est la mise en place d'un proxy applicatif. Ces opérations tronquent le trafic IP d'origine, génère de la charge sur les ressources nécessaires à la traduction à la réécriture du trafic dans un sens et dans un autre, ce qui n'est pas sans poser problème et génère un certain coût.

3.4. Transition IPv6

Il n'est plus envisagé de manière crédible traduire le nouveau protocole dans l'ancien, IPv6 dans IPv4.

Par contre, l'inverse, soit l'ancien dans le nouveau, IPv4 dans IPv6 annonce la prochaine étape de transition. Les solutions NAT64/DNS64 offrent la possibilité de déploiement "IPv6 Only".

3.5. NAT et pare-feu

On a tendance à confondre les fonctionnalités NAT avec celles du pare-feu.

Même si il est probable que le même logiciel prenne en charge les deux fonctions, ces procédures sont distinctes. Alors que le NAT permet de traduire le trafic, il ne protège en rien.

Cette fonction est prise en charge par le pare-feu à état qui arrête les connexions non sollicitées sur le réseau à protéger. Le proxy (mandataire) quant à lui aura d'autres fonction que la traduction telle que le contrôle du trafic qui lui est livré, avec des mécanismes de cache, d'authentification et de transformation du trafic dont aussi la traduction.

Dans tous les cas, c'est la fonction pare-feu qui protège des tentatives de connexions externes.

3.6. Adressage IPv4

Une adresse IPv4 est un identifiant de 32 bits représentés par 4 octets (8 bits) codés en décimales séparées par des points.

tableau adresses IPv4

Types d'adresses IPv4	Plages	Remarques
Adresses Unicast	`0.0.0.0` à `223.255.255.255`	Pour adresser les routeurs des réseaux d'extrémité et les ordinateurs accessible à travers l'Internet public
Adresses Unicast privées	`10.0.0.0/8` (Classe A), `172.16.0.0/12` (Classe B), `192.168.0.0/16` (Classe C)	L'objectif de départ des adresses IPv4 privées est d'identifier des connexions privées (sur des réseaux privés). Mais en situation de carence d'adresses IPv4 publiques, les adresses privées permettent aussi d'adresser des réseaux "clients" d'extrémité. Ceux-ci arrivent à placer du trafic sur l'Internet public grâce à des mécanismes de traduction d'adresses (NAT/PAT). Par définition, les adresses privées ne trouvent pas de destination sur l'Internet public.
Adresses Multicast	`224.0.0.0` à `239.255.255.255`	Ces adresses identifient plusieurs interfaces en général sur un réseau contrôlé.

Le masque de réseau lui aussi noté en décimal pointé indique avec les bits à 1 la partie réseau partagée par toutes les adresses d'un bloc. Les bits à 0 dans le masque indiquent la partie unique qui identifie les interfaces sur la liaison. Un masque de réseau est un suite homogène de bits 1 et puis de bits seulement à zéro. Il s'agit donc d'un masque de découpage de blocs homogènes d'adresses IP contigües. Ces blocs communiquent entre eux grâce à des "routeurs", sortent d'intermédiaire d'interconnexion, dont le rôle est justement de transférer du trafic qui ne leur est pas destiné. Les routeurs identifient un point géographique, lieux d'interconnexion des liaisons. Les routeurs IPv4 remplissent souvent des fonctions de traduction.

Par exemple, `192.168.1.25 255.255.255.0` indique un numéro de réseau (première adresse) `192.168.1.0` et un numéro de Broadcast `192.168.1.255`. Toutes les adresses comprises entre ces valeurs peuvent être utilisée par les interfaces attachées à une même liaison (un même switch).

Le masque peut aussi respecter la notation CIDR qui consiste à écrire le nombre de bits à 1 dans le masque. Soit dans l'exemple précédent une écriture de type `/24` pour les 24 bits à 1 (`255.255.255.0`). Le masque CIDR s'impose en IPv6 (imaginez des masques de 128 bits en hexadécimal). Cette notation est certainement plus intuitive et plus

simple à encoder ou à lire.

A cause du manque d'espace IPv4 disponible, on trouve souvent des masques qui chevauchent les octets, comme par exemple `/27` (`255.255.255.224`) qui offre 32 adresses (5 bits à zéro dans le masque) ou `/30` (`255.255.255.252`) qui offre 4 adresses (2 bits à zéro dans le masque), sans oublier la première (numéro du réseau) et la dernière (l'adresse de diffusion, *broadcast*) que l'on ne peut attribuer aux interfaces. On constate que si le bits à 1 dans le masque indiquent le découpage de cet espace codé sur 32 bits, les bits à zéro restants donnent l'étendue du bloc.

L'utilitaire `ipcalc` calcule pour nous les adresses IP :

```
ipcalc --help
```

```
Usage: ipcalc [OPTION...]
  -c, --check         Validate IP address for specified address family
  -4, --ipv4          IPv4 address family (default)
  -6, --ipv6          IPv6 address family
  -b, --Broadcast     Display calculated Broadcast address
  -h, --hostname      Show hostname determined via DNS
  -m, --netmask       Display default netmask for IP (class A, B, or C)
  -n, --network       Display network address
  -p, --prefix        Display network prefix
  -s, --silent        Don't ever display error messages

Help options:
  -?, --help          Show this help message
  --usage             Display brief usage message
```

Par exemple :

```
ipcalc -n -b -m 192.167.87.65/26
```

```
NETMASK=255.255.255.192
BROADCAST=192.167.87.127
NETWORK=192.167.87.64
```

On trouvera un utilitaire plus explicite avec `sipcalc`, il est en dépôt chez Debian/Ubuntu. Sous Centos/RHEL, il sera nécessaire de le compiler sois-même.

```
sipcalc -I ens33
```

```
-[int-ipv4 : ens33] - 0

[CIDR]
Host address               - 172.16.98.241
Host address (decimal)       - 2886755057
Host address (hex)        - AC1062F1
Network address              - 172.16.98.0
Network mask               - 255.255.255.0
Network mask (bits)         - 24
Network mask (hex)        - FFFFFF00
```

```
Broadcast address        - 172.16.98.255
Cisco wildcard                - 0.0.0.255
Addresses in network        - 256
Network range                - 172.16.98.0 - 172.16.98.255
Usable range                - 172.16.98.1 - 172.16.98.254
```

3.7. Adressage IPv6

Les adresses IPv6 sont des identifiants uniques d'interfaces codés sur 128 bits et notés en hexadécimal en 8 mots de 16 bits (4 hexas) séparés par des " :".

Par exemple, pour l'adresse `2001:0db8:00f4:0845:ea82:0627:e202:24fe/64` dans son écriture extensive :

```
2001:0db8:00f4:0845:ea82:0627:e202:24fe
---- ---- ---- ---- ---- ---- ---- ----
16b  16b  16b  16b  16b  16b  16b  16b
------------------- -------------------
    Préfixe          Interface ID
```

Ecriture

Voici l'écriture résumée :

```
2001:0db8:00f4:0845:ea82:0627:e202:24fe
2001:-db8:--f4:-845:ea82:-627:e202:24fe
2001:db8:f4:845:ea82:627:e202:24fe
```

Ou encore l'adresse `fe80:0000:0000:0000:bb38:9f98:0241:8a95` peut être résumée en `fe80::bb38:9f98:241:8a95`.

Configuration

Ces adresses peuvent être configurées :

- De manière automatique (autoconfiguration), sans état
- De manière dynamique avec serveur DHCPv6, avec état
- De manière automatique et de manière dynamique
- De manière statique

Une interface IPv6 peut accepter plusieurs adresses et dans des préfixes distincts. L'idée est d'améliorer les politiques de routage et de filtrage en fonction de ces adresses.

Adresses Unicas Globale et Link-Localt

Un premier exemple avec qui illustre des adresses Unicast :

```
# ip -6 a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 state UNKNOWN qlen 1
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 state UP qlen 1000
    inet6 2001:dc8:f4:845:ea82:627:e202:24fe/64 scope global noprefixroute dynamic
       valid_lft 1209585sec preferred_lft 604785sec
    inet6 fe80::bb38:9f98:241:8a95/64 scope link
       valid_lft forever preferred_lft forever
```

On y trouve deux interfaces :

- **`lo`** qui prend la seconde adresse de l'espace IPv6. `::1/128` ne joint qu'elle-même directement comme adresse de "Loopback".
- **`eth0`** : qui prend deux adresses IPv6 :
- L'une toujours présente se reconnait par son préfixe `fe80::/10`. Ces adresses sont uniques sur chaque interface et communiquent uniquement avec d'autres interfaces sur le lien local (des *voisins*) d'où leur "Link-Local Address".
- L'autre adresse avec le préfixe `2001:dc8:f4:845::/64` indique une adresse publique, soit routable, joignable par Internet et permettant de placer du trafic sur Internet. On l'appelle une adresse GUA "Gobal Unicast Address". Elles s'identifient dans le bloc `2000::/3`.

On notera que :

- Les adresses GUA et link-local disposent d'un masque `/64`. C'est le masque par défaut de toute interface IPv6 d'un point terminal (endpoint). On propose aux connexions d'entreprise le routage d'un bloc `/48` qu'elle pourrait découper sur les 16 bits suivant pour obtenir 65536 blocs `/64` à adresser sur ses réseaux.
- Les adresses GUA et link-local disposent de deux valeurs de durée de vie : `valid_lft` et `preferred` qui déterminent la durée de leur validité.

Adresses Mulitcast

Les groupes Multicast dans lesquels les interfaces sont inscrites.

```
# ip -6 maddress
1:        lo
        inet6 ff02::1
        inet6 ff01::1
2:        eth0
        inet6 ff02::1:ff02:24fe
        inet6 ff02::1:ff41:8a95
        inet6 ff02::1
        inet6 ff01::1
```

Cela signifie que les interfaces acceptent le trafic dont l'adresse de destination est une de ces adresses qui sont partagées par plusieurs interfaces sur plusieurs hôtes IPv6. A priori, le Multicast n'est pas transféré par les routeurs ; les commutateurs le traitent comme du *Broadcast* (Diffusion).

Ceci précisé, le Multicast IPv6 désigne finement le trafic sur base de la portée et d'une destination. `ff02::1` signifie "tous les hôtes sur le réseau local" ; `ff01::1` signifie "tous les hôtes sur le noeud local". Les adresses `ff02::1:ff02:24fe` `ff02::1:ff41:8a95` servent de destination au trafic Neighbor Discovery (voir plus bas) qui demande l'adresse physique de livraison inconnue d'une adresse IPv6 à joindre, donc connue d'avance. On

reconnait après le préfixe `ff02::1:ff/104` les 24 derniers bits `02:24fe` et `ff41:8a95` de chacune des deux adresses `2001:db8:f4:845:ea82:627:e202:24fe` et `fe80::bb38:9f98:241:8a95`. Il y aura une adresse Multicast différente pour chaque adresse GUA, ULA (Unique Local) ou link-local aux 24 derniers bits différents.

La table de routage indique que le trafic pour l'Internet (`default`) est livré à adresse link-local du routeur `fe80::22e5:2aff :fe1b :656a`.

```
# ip -6 route
2001:db8:f4:845::/64 via fe80::22e5:2aff:fe1b:656a dev ens33  proto ra  metric 100
fe80::22e5:2aff:fe1b:656a dev ens33  proto static  metric 100
fe80::/64 dev ens33  proto kernel  metric 256
default via fe80::22e5:2aff:fe1b:656a dev ens33  proto static  metric 100
```

Adresses Unique Local (Unicast)

Un second exemple révèle l'existence d'adresses privées IPv6 dans le préfixe `fd00::/8`, ici sur l'interface `eth0` :

```
# ip -6 a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 state UNKNOWN qlen 1
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 state UP qlen 1000
    inet6 fd00:101::1a8/128 scope global dynamic
       valid_lft 3021sec preferred_lft 3021sec
    inet6 fe80::5054:ff:fe53:c52c/64 scope link
       valid_lft forever preferred_lft forever
```

Le préfixe `fd00::/8` nous renseigne un bloc d'adresses privées dont les 40 bits suivants sont censés être générés de manière aléatoire, ce qui est une obligation mais qui selon moi reste impossible à vérifier. Le préfixe suggéré offre alors un bloc `/48` dont on peut compenser 16 bits pour offrir 65536 blocs `/64`.

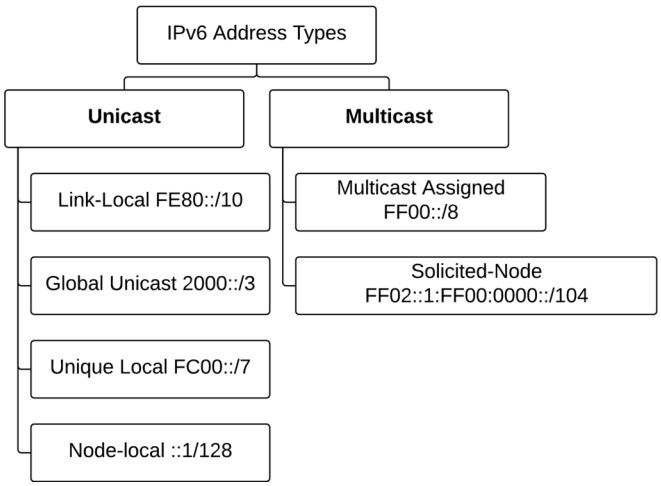
Ce type d'adresse privée à préfixe unique répond au problème des blocs IPv4 privés qui sont partagés par l'ensemble des postes de travail ou des périphériques mobiles dans le monde. Cet état de fait en IPv4 rend les connexions de bout en bout complexes à établir à travers un Internet public (notamment à travers des réseaux VPN, pour du partage peer-to-peer pour de la téléphonie Internet, etc.). Ce ne sont pourtant pas les solutions qui manquent mais celle-ci ne font que renforcer la mise en place de bricolages qui dégradent la connectivité.

Transition IPv6

L'Internet des Objets et les services en nuage annoncent une croissance exponentielle des besoins en connectivité auxquels seul IPv6 pourra répondre. L'Internet est entré en juin 2012 dans une très longue phase de transition duale d'IPv4 à IPv6. On pourrait encore parler d'IPv4 d'ici 2025 alors qu'IPv6 dominera les communications TCP/IP. Plusieurs explications apportent de l'eau au moulin de cette longue phase : une répartition des coûts sur tous les acteurs ; malgré son aspect logique une migration d'infrastructure est nécessaire à l'échelle globale, sans compter les serveurs publics qui ne seront jamais migrés en IPv6 et qu'il faudra maintenir en IPv4 …

Synthèse sur les adresses IPv6

En résumé pour IPv6, on peut retenir ce schéma :



Adresses IPv6

Etant donné la multiplication des sources et des destinations potentielles qu'offrent le protocole IPv6, on sera attentif aux configurations des pare-feux.

4. Protocoles de résolution d'adresses et de découverte des hôtes

- Afin d'encapsuler un paquet IP dans une trame, l'hôte d'origine a besoin de connaître l'adresse physique (MAC) de la destination.
- En IPv4, c'est le protocole ARP (Address Resolution Protocol) qui remplit cette fonction. Les hôtes IPv4 maintiennent une table appelée cache ARP.
- En IPv6, c'est le protocole ND (Neighbor Discovery), sous-protocole IPv6, qui reprend cette fonction. Les hôtes IPv6 maintiennent une table appelée table de voisinage.

4.1. Commandes utiles

- Table ARP sous Windows et Linux : `arp -a`
- Table ARP sous Linux : `ip -4 neigh`
- Table de voisinage sous Linux : `ip -6 neigh`
- Table de voisinage sous Windows : `netsh interface ipv6 show neighbors`

4.2. ARP (Address Resolution Protocol)

Au moment de l'encapsulation d'un paquet IPv4 dans une trame Ethernet ou Wi-fi, l'hôte émetteur connaît d'avance l'adresse IP de destination. Mais comment peut-il connaître son adresse physique correspondante (l'adresse MAC de destination par exemple) afin de placer le trafic sur le support ?

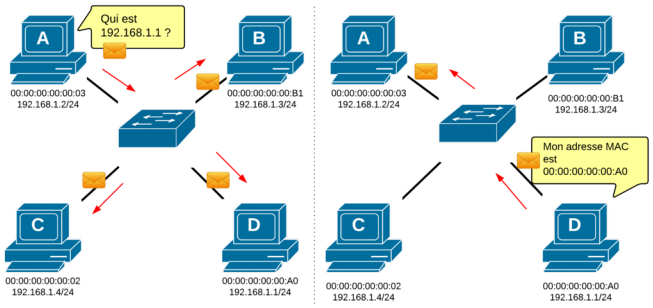
Un hôte TCP/IP ne peut connaître l'adresse de destination sans qu'elle ne s'annonce elle-même de *manière gratuite* ou de *manière sollicitée*.

Dans le but de maintenir une correspondance entre des adresses IP à joindre et leur adresses physiques de destination, les hôtes TCP/IP entretiennent une "**table ARP**" pour les adresses IPv4 et une "**table de voisinage**" pour les adresses IPv6.

ARP est un protocole indépendant d'IPv4 qui offre ce service de résolution d'adresses.

En IPv6, ce sont des paquets ICMPv6 appelés Neighbor Discovery (ND) qui sont utilisés selon un mode sensiblement différent. En IPv6, les fonctions d'informations et de contrôle (ICMPv6) ont été améliorées et renforcées.

- La requête ARP émane en Broadcast et la réponse est envoyée en unicast. ND (IPv6) aura un fonctionnement similaire en utilisant une adresse Multicast spéciale en lieu et place du Broadcast.

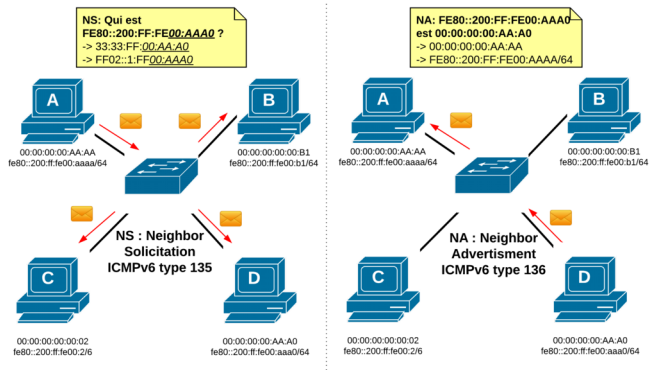


ARP (Address Resolution Protocol)

- Capture : https ://www.cloudshark.org/captures/e64eaac12704?filter=arp

4.3. ND (Neighbor Discovery)

- Découverte de voisin sollicitée



ND (Neighbor Discovery)

- Capture : https ://www.cloudshark.org/captures/85556fc52d28

5. Protocole de résolution de noms

- Au niveau protocolaire, seuls les adresses IP sont utilisées pour déterminer les partenaires d'une communication.
- Mais dans l'usage courant d'Internet, on utilise des noms pour joindre des machines sur le réseau : c'est plus facile à manipuler que des adresses IP.
- Le protocole et le système DNS permet de résoudre des noms en adresses IP.
- DNS est une sorte de service mondial de correspondance entre des noms et des adresses IP. DNS utilise le port UDP 53.

Recherche DNS

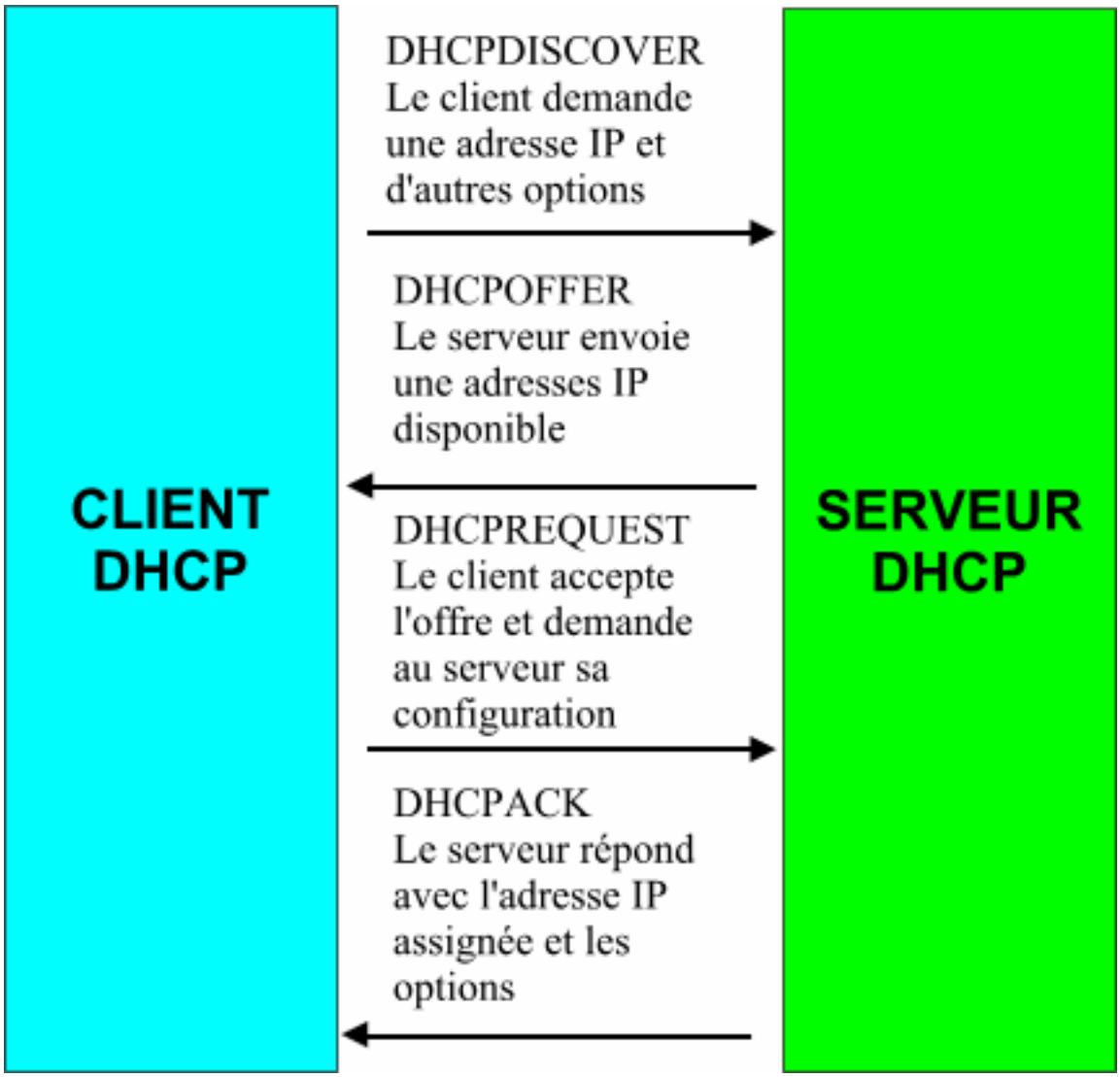
Source : https ://visual.ly/help-understanding-dns-lookups

6. Protocoles d'attribution d'adresses

- Avant de pouvoir émettre du trafic TCP/IP, une interface doit disposer au minimum d'une adresse IP et de son masque et, éventuellement d'autres paramètres (passerelle par défaut, résolveur DNS, etc.).
- En IPv4, c'est DHCP qui permet d'attribuer ces paramètres à une interface qui le demande. DHCP maintient un état des adresses attribuées par un mécanisme de bail (à durée déterminée).
- En IPv6, le comportement par défaut est l'autoconfiguration des interfaces mais la version actualisée de DHCPv6 fournit un service géré des adresses.

6.1. DHCP (IPv4)

- La procédure d'attribution d'adresses en DHCP (IPv4) consiste en l'échange de 4 messages sur les ports UDP 67 et 68.
- Le premier message DHCP émane du client en Broadcast.



Echange entre client et serveur DHCP

Capture de trafic DHCP : https ://www.cloudshark.org/captures/c109b95db0af

Dans un session typique, le client diffuse (Broadcast) un message DHCPDISCOVER sur son segment local. Le client peut suggérer son adresse IP et la durée du bail (lease). Si le serveur est sur le même segment, il peut

répondre avec un message DHCPOFFER qui inclut une adresse IP valide et d'autres paramètres comme le masque de sous-réseau. Une fois que le client reçoit ce message, il répond avec un DHCPREQUEST qui inclut une valeur identifiant le serveur (pour le cas où[1] il y en aurait plusieurs). Cette valeur l'identifie de manière certaine et décline implicitement les offres des autres serveurs. Une fois le DHCPREQUEST reçu, le serveur répond avec les paramètres définitifs de configuration par un message DHCPACK (si le serveur a déjà assigné l'adresse IP, il envoie un DHCPNACK).

Si le client détecte que l'adresse IP est déjà utilisée sur le segment, il envoie un DHCPDECLINE au serveur et le processus recommence.

Si le client reçoit un message DHCPNACK du serveur après un DHCPREQUEST, le processus recommence également.

Si le client plus besoin d'une adresse IP, il envoie un DHCPRELEASE au serveur.

Si le client veut étendre la durée du bail qui lui est allouée, il envoie un DHCPREQUEST au serveur dans lequel le champ 'ciaddr' correspondra à son adresse IP actuelle. Le serveur répondra avec un DHCPACK comprenant la nouvelle durée du bail.

6.2. DHCPv6

Cette matière est abordée dans le chapitre Services d'infrastructure.

7. Interaction des protocoles

- Avant qu'une interface puisse envoyer du trafic faut-il :
- qu'elle ait obtenu une adresse IP statique ou dynamique (DHCP en IPv4 ou autoconfiguration/DHCPv6 en IPv6) ;
- qu'elle ait résolu le nom de l'hôte destinataire en adresse IP (DNS sur IPv4 ou sur IPv6) ;
- qu'elle ait obtenu l'adresse du livraison physique de la destination locale ou de la passerelle par défaut si la destination n'est pas locale (ARP en IPv4 ou ND en IPv6).

8. Autres protocoles de gestion

D'autres protocoles de gestion importants se rencontreront dans les réseaux TCP/IP, à titre d'exemples :

- ICMP qui permet en IPv4 et en IPv6 d'obtenir du diagnostic IP (`ping` et `traceroute`).
- NTP qui permet de synchroniser les horloges des hôtes sur le réseau.
- SNMP qui permet de collecter des informations sur le matériel à travers le réseau.
- SSH qui permet de monter une console distante à travers TCP/IP.
- Le routage IP met en oeuvre du NAT sur les passerelles des réseaux privés pour offrir une connectivité à l'Internet.
- …

Révisions

Page vide.