Gestion d'un serveur personnel

L'art d'héberger moultes services sécurisés sur un serveur personnel

Version 0.1

---

lemanchotvolant.com
Julien VAUBOURG <julien@vaubourg.com>

5 avril 2010

Table des matières

1. Préambule

1.1 Objectif

Ce document accompagne le `P.R.A.`[1] du *Manchot volant*. Il peut également servir de base pour la conception d'un serveur personnel — entendre une seule IP et une seule machine — destiné à héberger de nombreux services web. L'utilisation de technologies professionnelles récentes — et libres — permettront de garantir une organisation des accès au serveur puissante ainsi qu'une souplesse dans l'administration des services tout en garantissant une sécurité solide. Cette dernière étant indispensable compte tenu de la quantité de données qui seront présentes sur une même machine et le nombre de possibilités d'accès extérieurs nécessaires.

1.2 Un point sur le matériel

Le serveur du Manchot volant est un petit serveur *rackable* 3U, utilisé habituellement en entreprise pour des solutions de type `U.T.M.`[2] (voir les photos en annexe A et B).

Spécifications :
- 1 carte mére mini-ITX LV-670 Series
- 1 Intel(R) Celeron(R) CPU 2.40GHz
- 256 Mo de RAM
- 2 disques durs IDE de 80 Go
- 1 alimentation de 220 Watt
- 1 port série branché en PCI

Périphériques :
- Un onduleur APC Backups CS 250
- Un disque dur SATA 1,5 To sur dock USB (la carte ne supporte pas le SATA, et le boîtier est plein)
- Ce qui ressemble à une alimentation d'ordinateur portable, en réalité un terminal modem GSM Siemens M20 branché en série

2. Installation du système

2.1 Apéritif

L'installation se fait à l'aide d'un lecteur DVD externe branché en USB et d'un CD-ROM `netinstall` de Debian est utilisé. Ce type de CD-ROM ne contient que le minimum nécessaire pour guider dans l'installation, partitionner les disques, et télécharger l'intégralité des fichiers de la distribution par Internet.

La langue choisie est l'anglais : plus de réponses pour les messages d'erreur et économie des paquets de langue.

Language English
Country Other / Europe / France
Keymap French
Hostname manchot
Domain name lemanchovolant.com

[1] Plan de Reprise d'Activités
[2] *Unified Threat Management* : La machine a été acheté à une entreprise qui se sert de ce type de matériel pour concevoir des solutions tout-en-un de sécurité. Destinées à faire pare-feu sur le frontal les réseaux d'entreprise clientes, elles ont été conçues pour une utilisation intensive sans relâche.

Une fois arrivé à l'étape *Partitionning method*, choisir *Manual*.
Commencer par supprimer toutes les partitions des deux disques dur.

2.2 Configuration du RAID 1

2.2.1 Préambule

Le `RAID 1` est une technologie permettant d'écrire simultanément sur deux disques durs à la fois, assurant ainsi la pérennité des données.

Pour configurer du `RAID 1`, il faut :

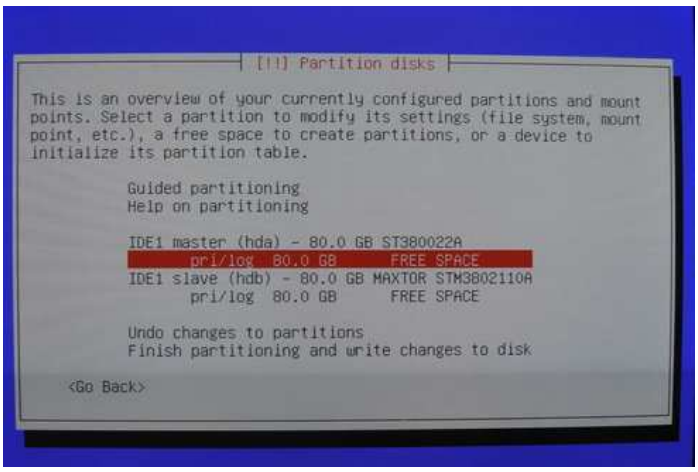
1. Créer des partitions de façon identique sur deux disques durs
2. Définir ces partitions comme étant des partitions de `RAID`
3. Utiliser l'outil de gestion du `RAID` de `partman` (le partitionneur de Debian) pour indiquer les partitions `RAID` qui devront être regroupées en « périphérique `RAID 1` » (soit une partition virtuelle qui en regroupe deux physiques, avec l'espace d'une seule), pour ne former plus qu'une partition visible

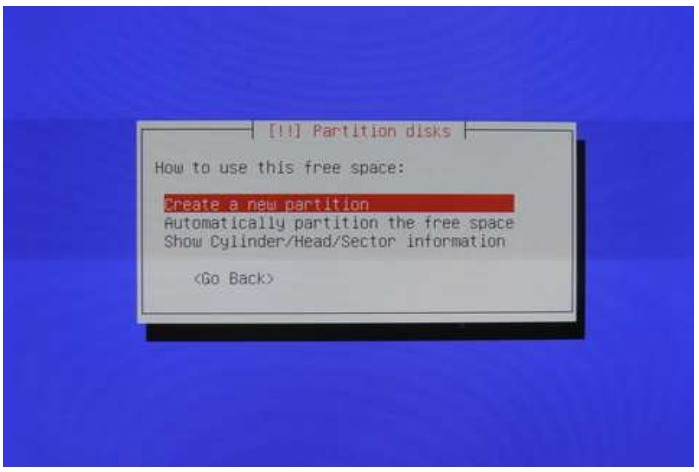
Les disques durs seront rassemblés en deux partitions/périphériques `RAID 1` : une petite pour le `boot` (correspondant au dossier */boot* du système et contenant le noyau Linux) et une grosse pour tout le reste des données. Ceci s'explique par l'utilisation par la suite d'un système de fichiers pour le système qui n'est pas reconnu par la plupart des `boot loader`. Il est donc nécessaire de séparer le disque en deux partitions avec deux systèmes de fichiers différents.

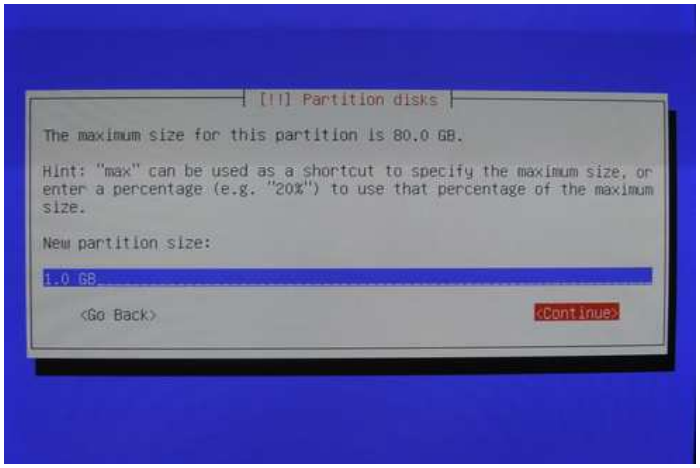
Toutes les étapes ont été détaillées dans la série de miniatures suivante.

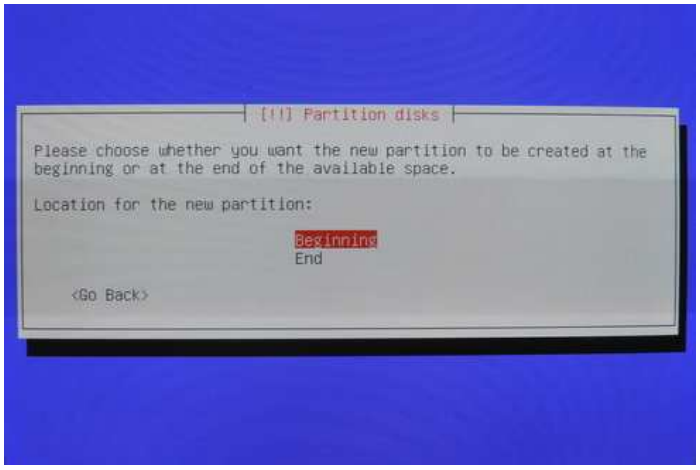
2.2.2 Création des périphériques `RAID 1`

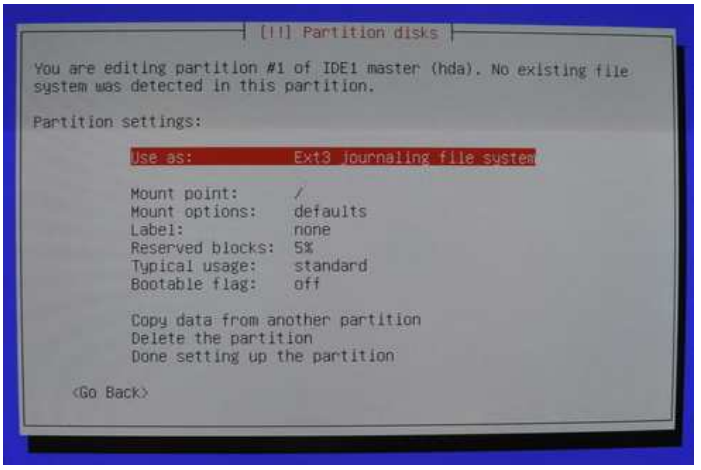
1. Commencer par créer une première partition `RAID` sur le premier disque dur :



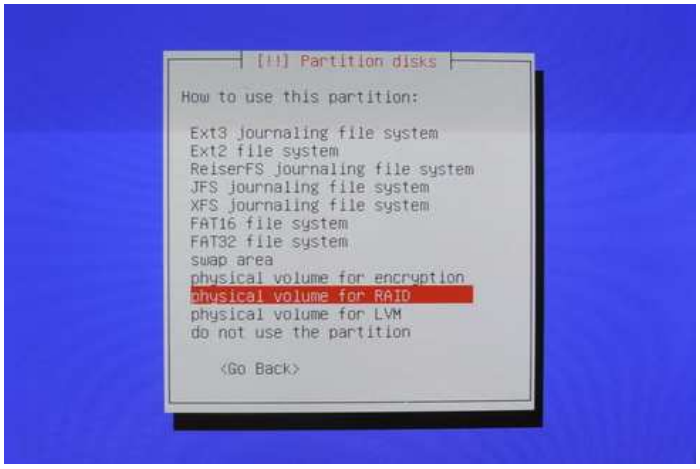


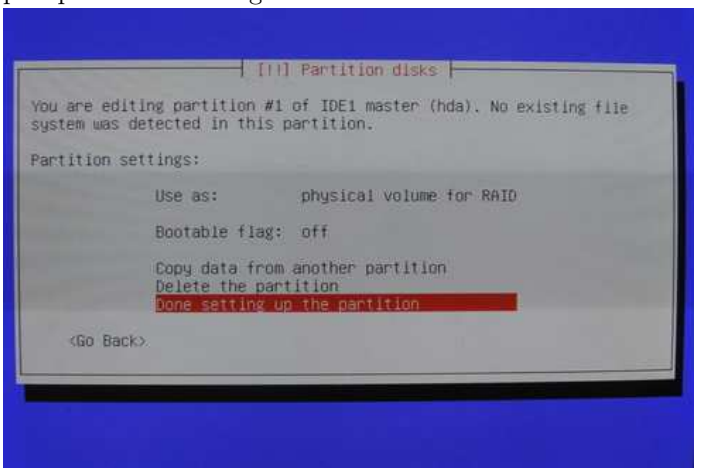


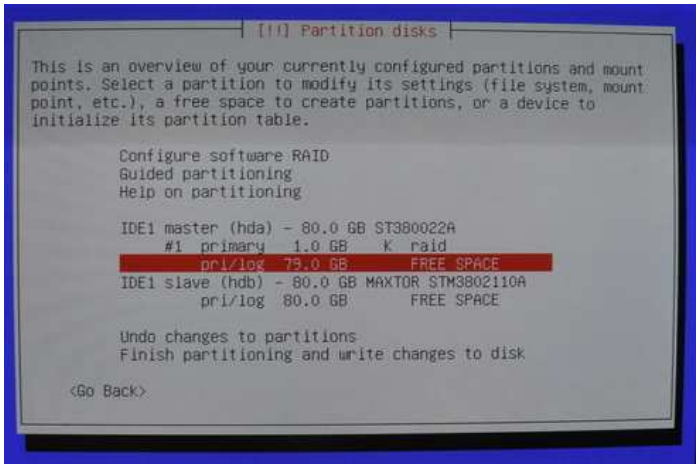




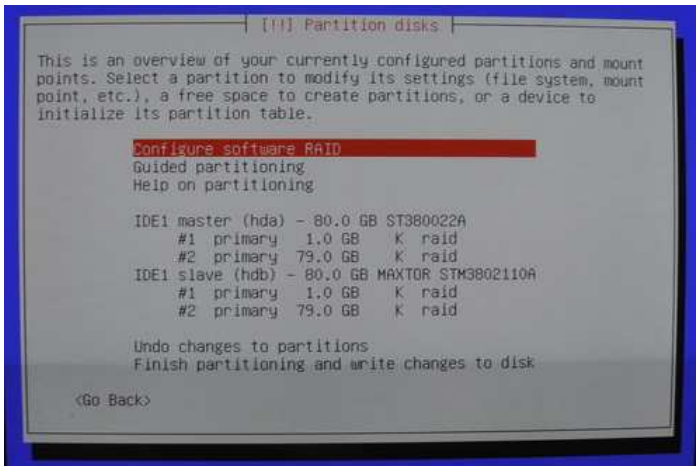
Le `Use as` peut indiquer une autre valeur, peu importe puisqu'il faut la changer.

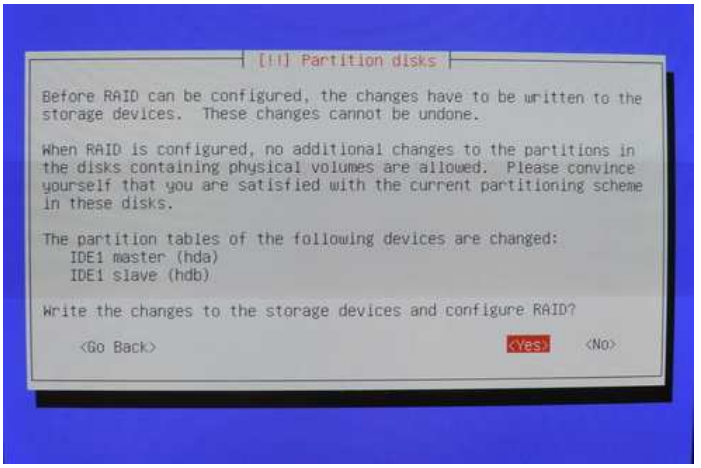


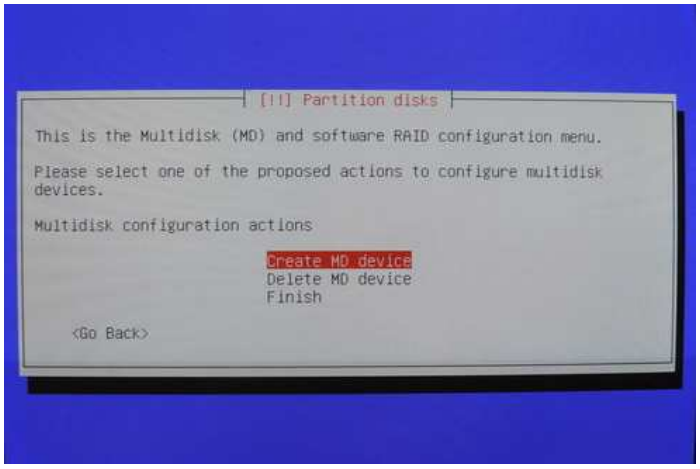


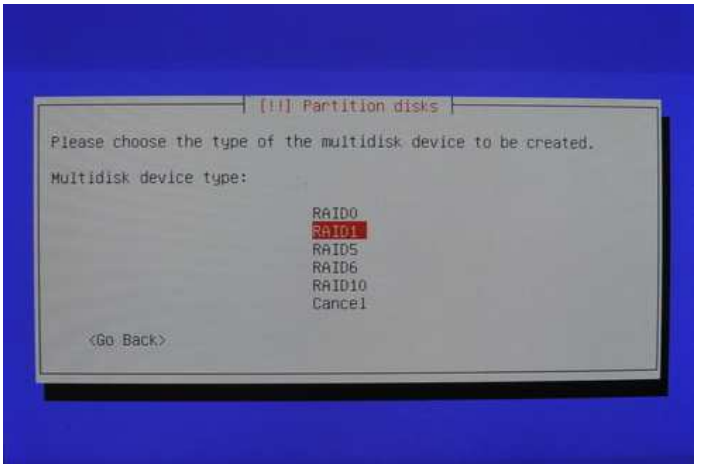


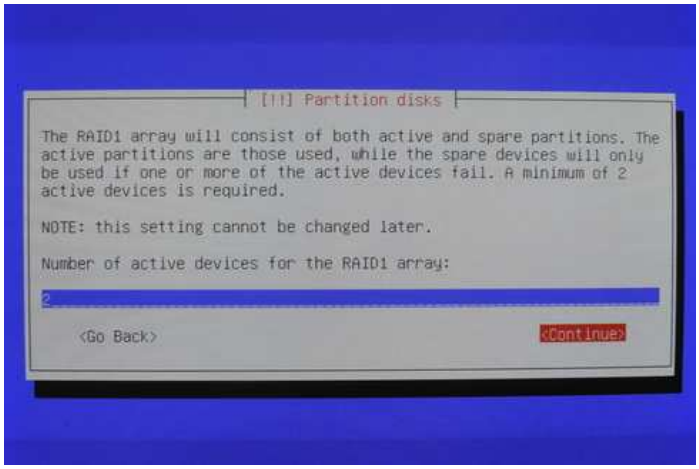
2. Reprendre la procédure 1 pour créer une seconde partition sur ce premier disque, en prenant cette fois ci tout l'espace disponible restant (valeur par défaut pour la taille).
3. Reprendre la procédure 1 puis 2 à l'identique, mais pour le second disque dur.
4. Créer les partitions (périphériques) `RAID 1` à partir des partitions `RAID` créées :

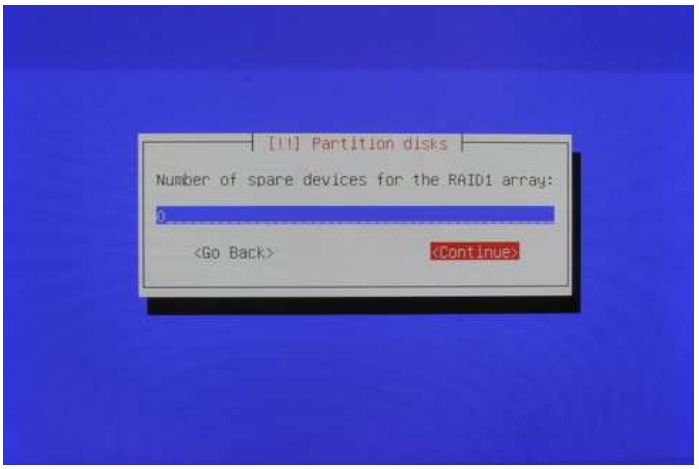


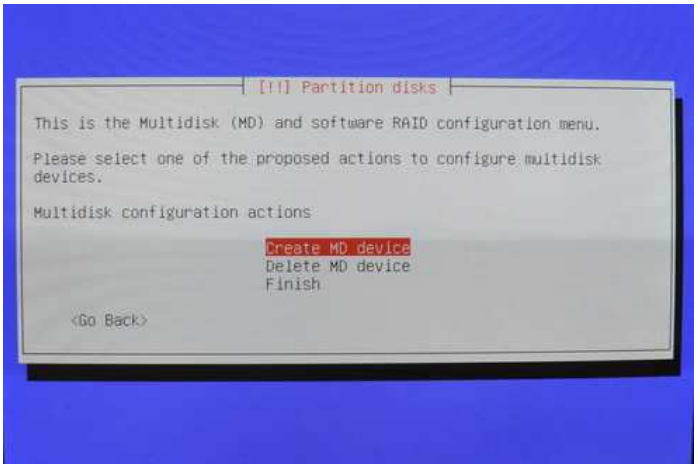






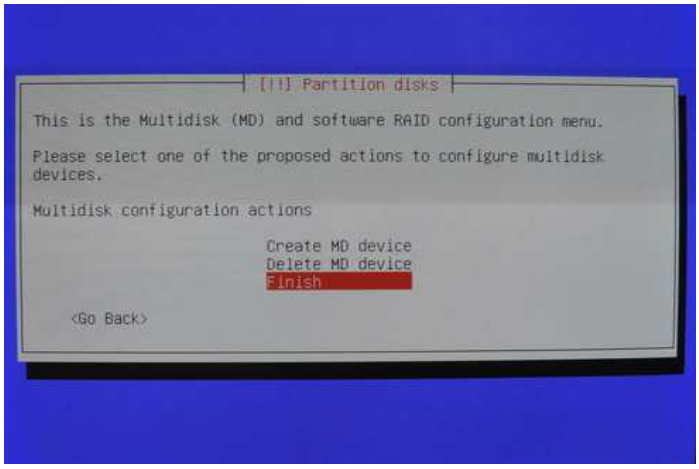






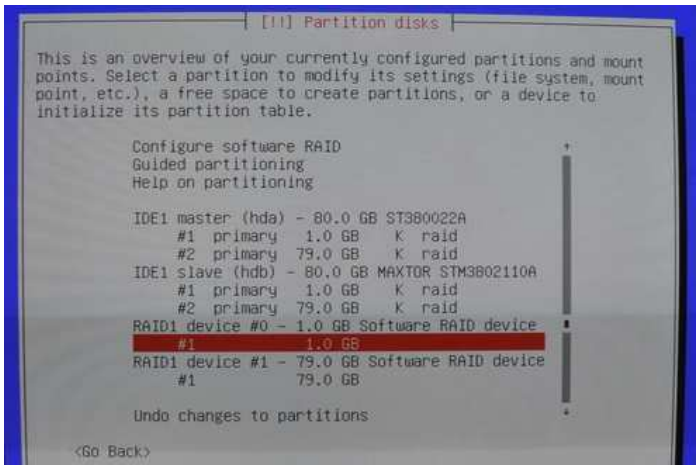
Sélection des partitions identiques sur les deux disques durs, paire à paire. Lors de la création du second volume `RAID`, seuls les deux autres seront proposés.

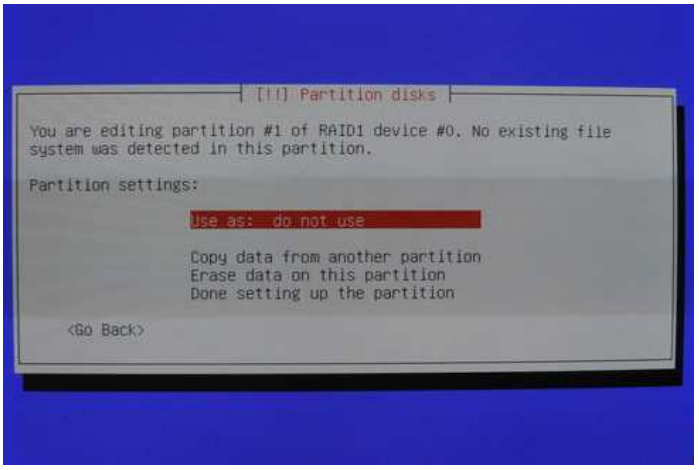
5. Reprendre la procédure 4 (à partir du *Create MD device*) en sélectionnant cette fois ci les deux partitions identiques restantes
6. Terminer la configuration du `RAID` :

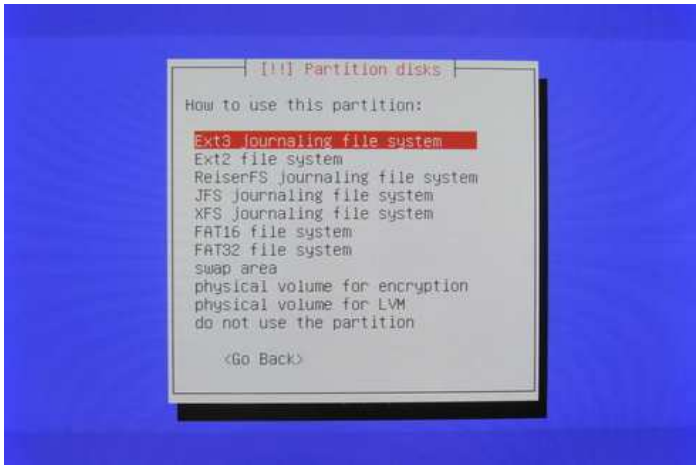


2.2.3 Systèmes de fichier des partitions

1. Définir le système de fichiers du premier périphérique `RAID 1` (`boot`) comme étant de l'`ext3` :







2. Reprendre la procédure en 1 pour définir le système de fichier du second périphérique raid comme étant l'équivalent d'un volume `LVM` (*physical volume for LVM*)

2.3 Partitionnement avec LVM

2.3.1 Préambule

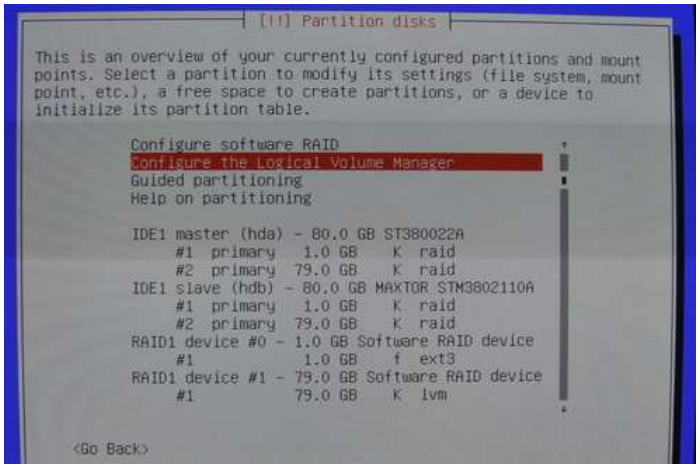
Le système de fichiers `LVM` permet de créer des partitions logiques sur une seule partition physique (de type `LVM`). A l'inverse des partitions physiques, ces partitions seront très aisément modulables par la suite. Il sera ainsi aisé d'en redimensionner pour en recréer de nouvelles, et c'est principalement pour cet intérêt qu'elles seront utilisées.

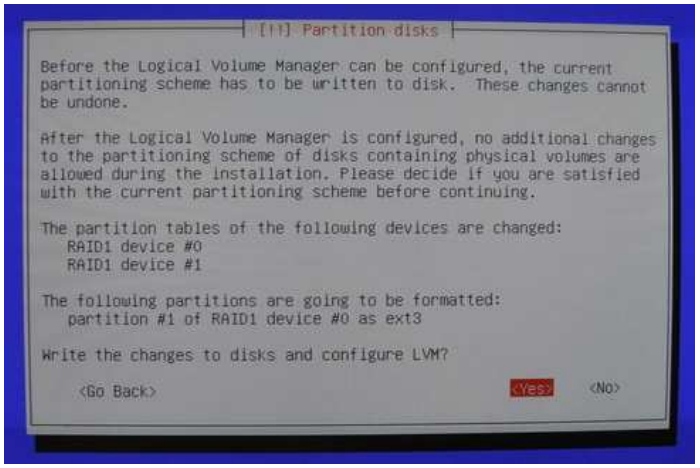
Avant de créer une partition logique, il faut créer un groupe logique (*volume group* ou encore `vg`). L'intérêt de tels groupes et de pouvoir créer des partitions logiques à cheval sur deux partitions physiques de type `LVM` (*physical volume*, ou encore `pv`), possibilité qui ne sera pas exploitée. Au sein d'un groupe, il est alors possible de créer des partitions logiques (*logic volume* ou encore *lv*).

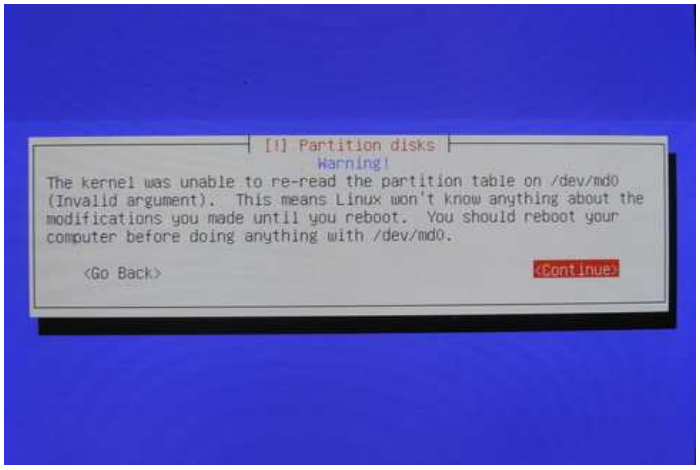
Lors de l'étape précédente, indiquer le système de fichiers `LVM` pour une des partitions `RAID 1` a créé un `pv`.

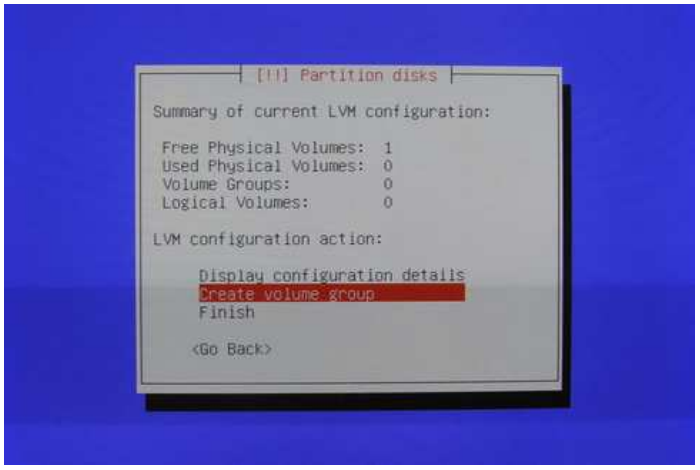
2.3.2 Création d'un vg

Suivre les miniatures suivantes :

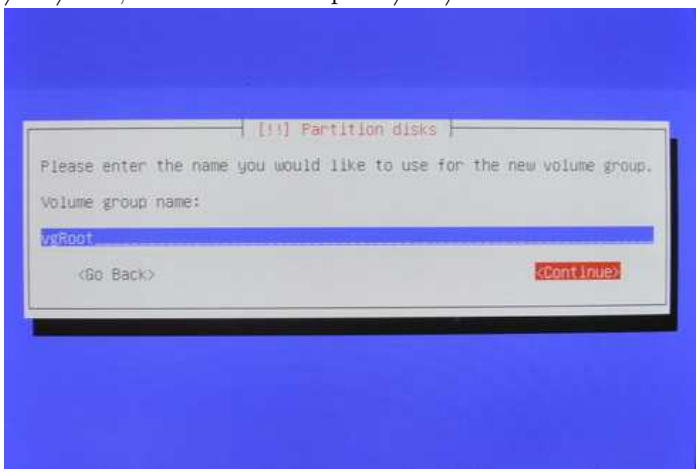


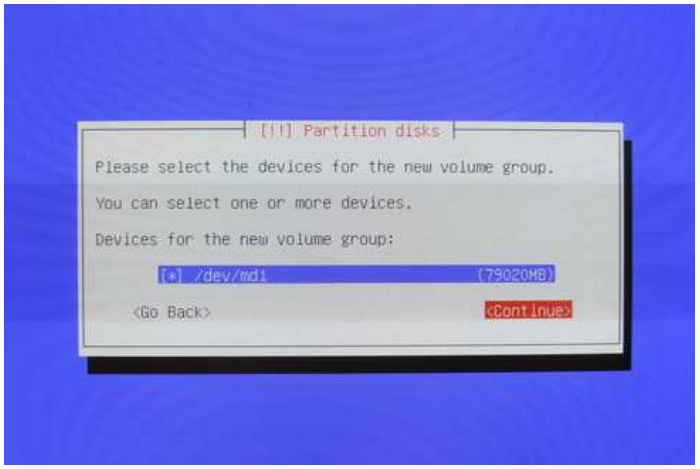






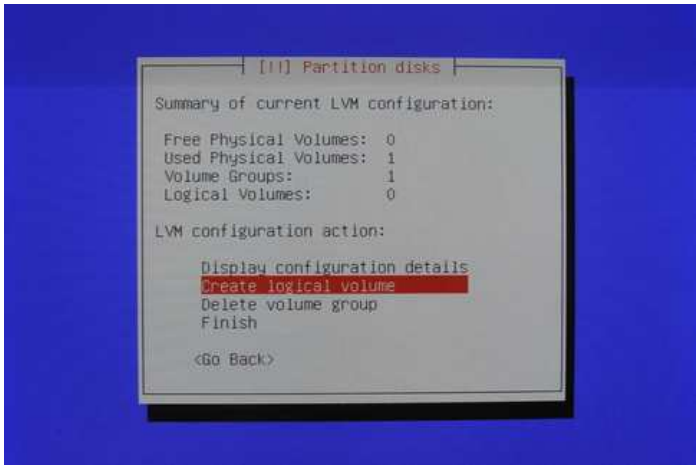
Cette avertissement peut apparaître deux fois : une fois pour */dev/md0*, une seconde fois pour */dev/md1*.

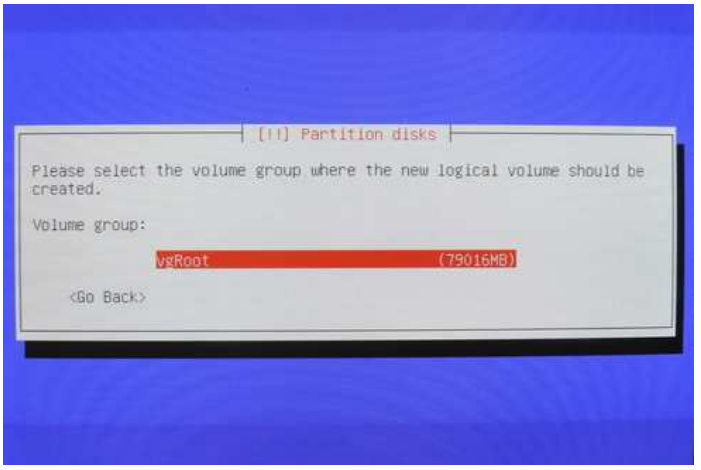


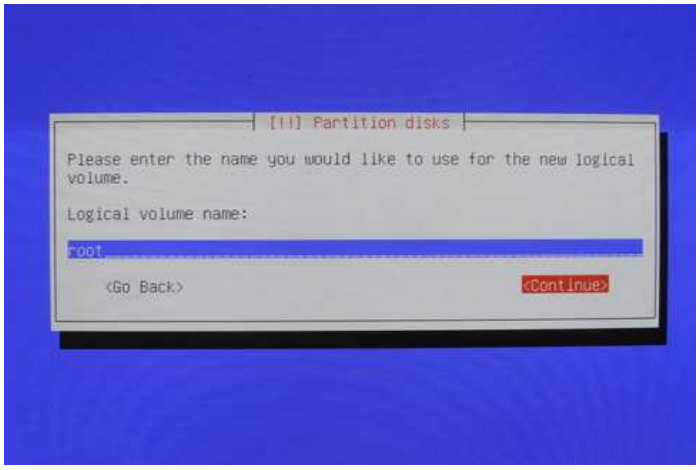


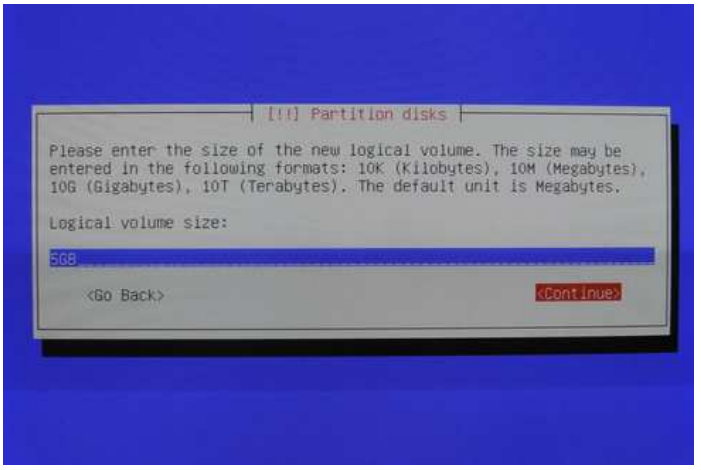
2.3.3 Création des `lv` (partitions)

1. A partir du menu d'arrivée de l'étape précédente, créer une première partition logique (`root`, qui correspondra à la racine `/` du système) :



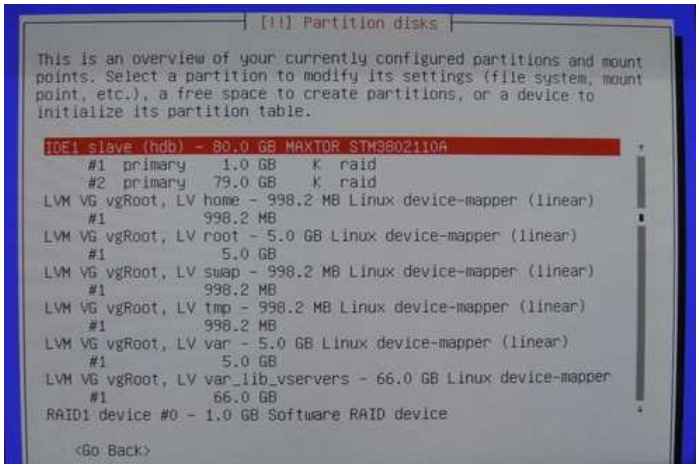






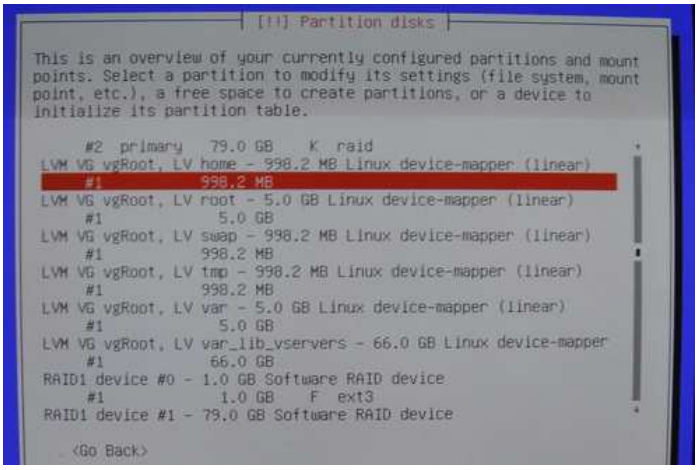
2. Reprendre la procédure en 1 avec chacune des partitions à créer (même dans le cas de gros disques, la taille des cinq premières partitions est largement suffisante) :
   - `root` d'une taille de 5GB **(déjà créée)**
   - `home` d'une taille de 1GB
   - `tmp` d'une taille de 1GB
   - `swap` d'une taille de 1GB
   - `var` d'une taille de 5GB
   - `var_lib_vservers` de la taille restante (valeur par défaut proposée)

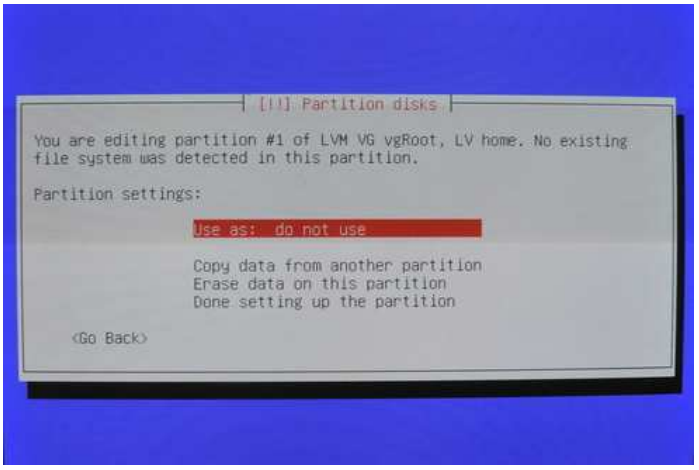
   NOTE : Dans le cadre d'une *reprise d'activités*, c'est également le moment opportun pour créer les différentes partitions correspondantes au VServers (technologie qui sera abordée par la suite) qui seront recréés lors de la procédure de restauration automatique. Nommer les partitions selon le format `var_lib_vservers_`*`nom`*.
3. Le résultat devrait ressembler à ceci :

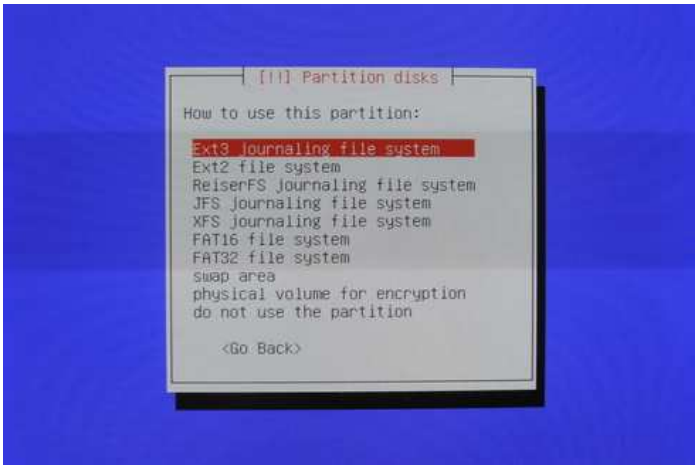


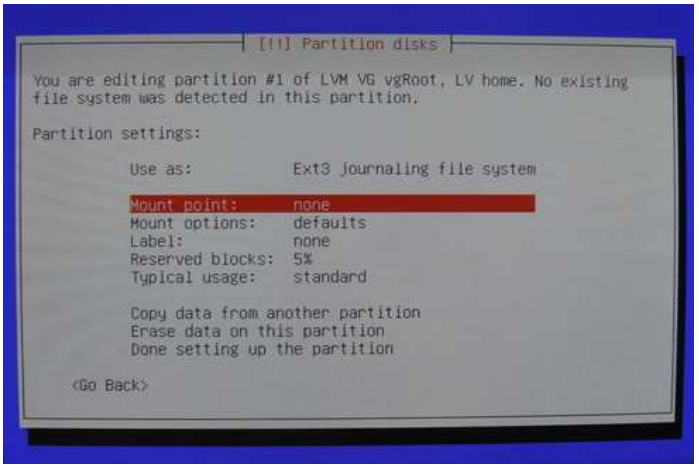
2.3.4 Systèmes de fichiers et points de montage des `lv`

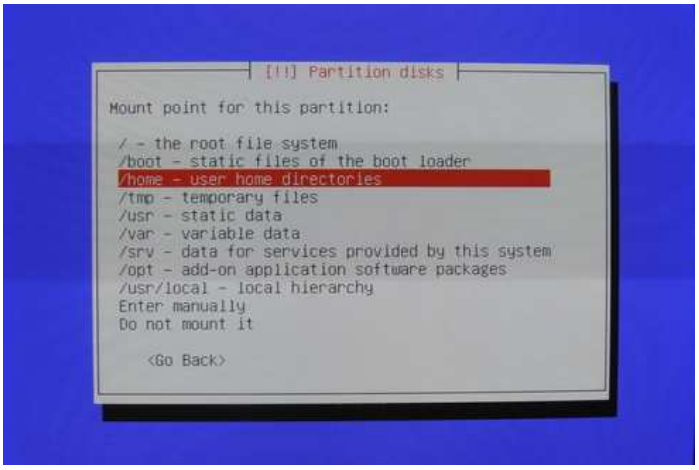
1. Pour chacun des `lv` créés, il faut définir un système de fichier et le point de montage dans le système qui lui correspondra :

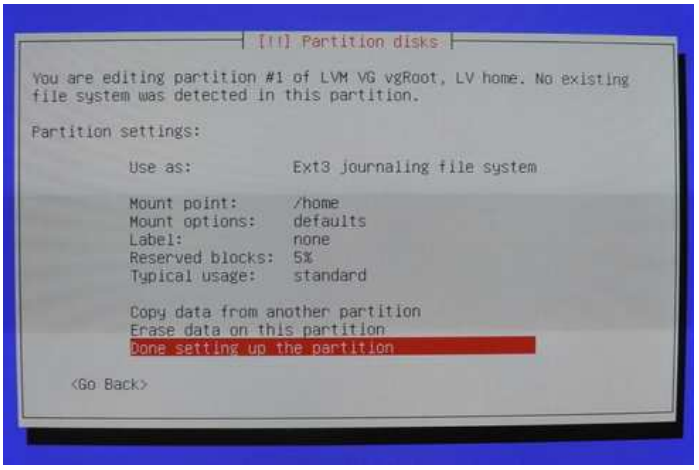










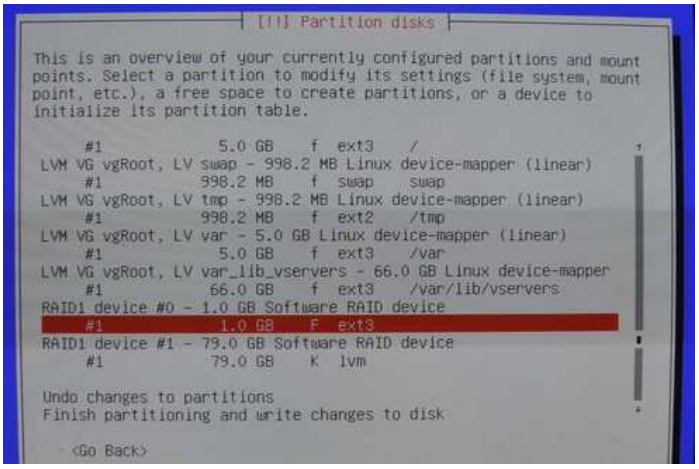


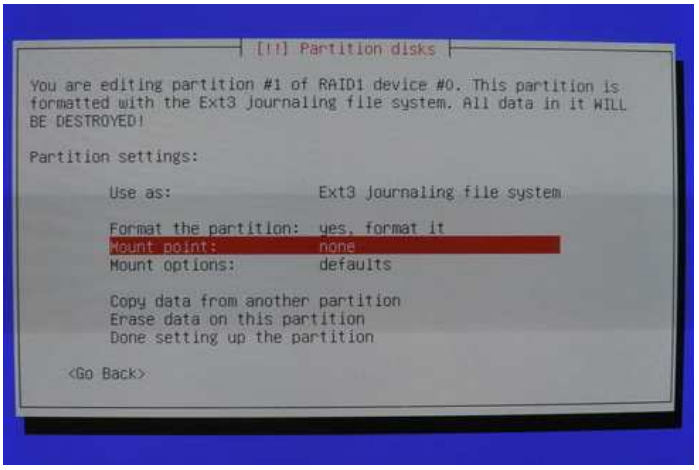
2. Reprendre la procédure en 1 pour chacune des partitions/`lv` (dans l'ordre d'apparition) :
   - `home` de type `ext3` à monter sur */home* **(déjà fait)**
   - `root` de type `ext3` à monter sur */*
   - `swap` de type `swap area`, sans point de montage
   - `tmp` de type `ext2` à monter sur */tmp*
   - `var` de type `ext3` à monter sur */var*
   - `var_lib_vservers` de type `ext3` à monter sur */var/lib/vservers* (choisir *Enter manually* tout en bas pour écrire le chemin)

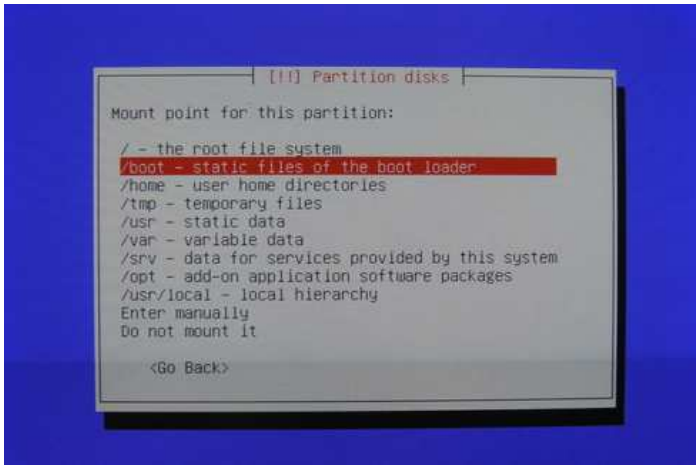
   NOTE : Dans le cadre d'une *reprise d'activités*, configurer les partitions supplémentaires avec les points de montage `/var/lib/vservers/`*`nom`*.

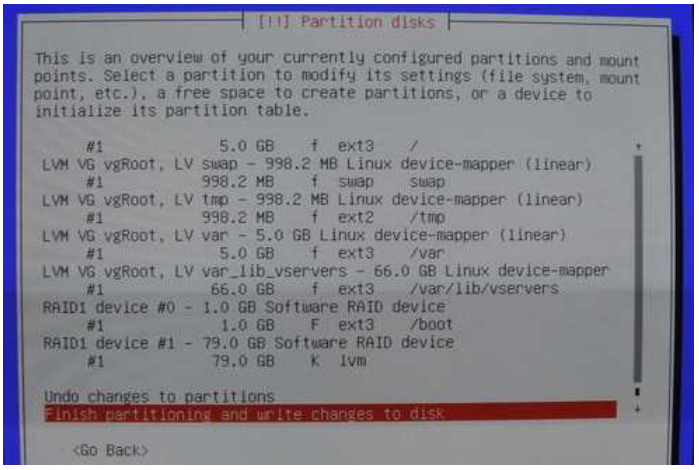
2.4 Point de montage du `boot` et validation

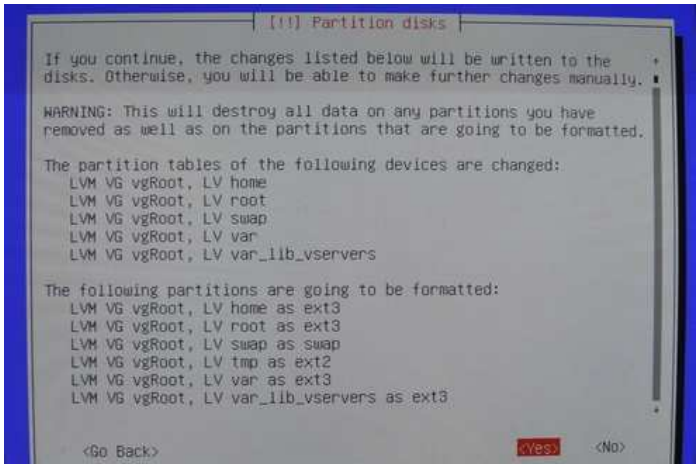
Attribuer le point de montage de la seconde partition `RAID 1` à */boot* et finaliser le partitionnement :





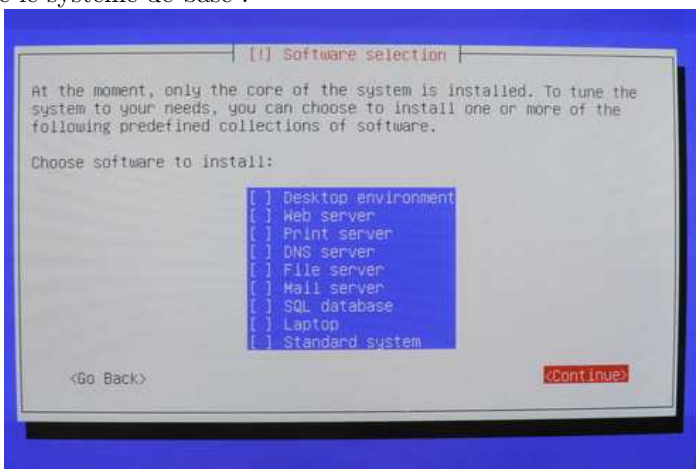






2.5 Dessert

La suite se fait tout seul :
- Mot de passe d'une vingtaine de caractères pour `root`
- Création d'un utilisateur `auk` dont le mot de passe importe peu
- Dépôts de paquets en France, miroir *ftp2.fr.debian.org*
- Pas de `HTTP Proxy`
- Participation au *Popularity Contest* (parce que le *popcon* c'est cool)
- Ne rien cocher, pas même le système de base :



– Et bien entendu, installation de `Grub`
– Enjoy ! :o)

3. Configuration du système

3.1 Installation et configuration de SSH

Exécuter :

```
apt-get install ssh
```

Points remarquables de la configuration (*/etc/ssh/sshd_config*) :

```
Port 42
ListenAddress 192.168.0.200

PermitRootLogin no
PasswordAuthentication no
AuthorizedKeysFile %h/.ssh/authorized_keys

AllowUsers auk
```

Le port SSH sera le port 42, et l'accès ne sera possible que pour l'utilisateur `auk`, à l'aide d'une clé SSH. Préciser l'IP du serveur, même si il ne possède qu'une carte réseau, a son importance par rapport à la suite de la configuration. Le fichier complet est disponible en annexe D.

Relancer le serveur :

```
/etc/init.d/ssh restart
```

Importer une clé publique pour `auk` (éventuellement par clé USB) :

```
su auk
mkdir /home/auk/.ssh
wget -O -\
 http://julien.vaubourg.com/files/julienVaubourg_sshKey.pub > /home/auk/.ssh/authorized_keys
```

Profiter de l'occasion pour supprimer le mot de passe de `auk` (qui rendra ainsi impossible l'utilisation de cet utilisateur sans passer par l'accès SSH) :

```
passwd auk -d
```

Fermer la session et se connecter au serveur depuis son canapé pour la suite de la configuration.

Pour ce faire, ajouter à son *.ssh/config* :

```
Host lmv
        HostName lemanchotvolant.com
        User auk
        Port 42
        IdentityFile %h/.ssh/id_dsa_ju
```

Puis :

```
ssh lmv
su
```

Ou bien directement :

```
ssh -p 42 -i .ssh/id_dsa_ju auk@lemanchotvolant.com
su
```

3.2 Configuration de base

3.2.1 Adresse statique

Attribuer une adresse statique au serveur :

```
sed\
 's/iface eth0 inet dhcp/iface eth0 inet static\naddress 192.168.0.200\nnetmask 255.255.255.0/'\
 -i /etc/network/interfaces
```

Dans le cas du routeur D-Link DIR-300, il n'est plus possible de se connecter en local au serveur depuis l'IP publique si l'IP privée du serveur n'a pas été attribuée par `DHCP`. Toutefois, si le système est dépendant du DHCP, il arrive parfois que la réponse de celui-ci arrive après que le service SSH soit lancé. Et parce que SSH écoute sur une adresse explicite (voir plus bas), il échouera au démarrage si cela se produit. Ainsi, pour ne pas avoir à gérer deux méthodes de connexions au serveur selon que l'on soit en local ou à distance, et pour n'avoir aucun risque de se retrouver à plusieurs kilomètres du serveur lorsque le SSH plantera au redémarrage, il peut être utile de forcer un DHCP — inutile — après que SSH se soit lancé en prenant en compte l'adresse définie statiquement. On prendra gare à indiquer au routeur d'attribuer toujours la même adresse IP à la carte réseau du serveur, afin que le DHCP ne change rien à la configuration.

Le script se trouve en annexe C.

Pour l'installer, le placer dans */etc/init.d/* puis exécuter :

```
chmod +x /etc/init.d/dhcp
update-rc.d dhcp defaults
```

3.2.2 Vérifier l'installation

Pour vérifier que le `RAID` est bien fonctionnel et qu'un disque dur n'a pas failli :

```
# cat /proc/mdstat
Personalities : [raid1]
md1 : active raid1 hda2[0] hdb2[1]
      77168128 blocks [2/2] [UU]

md0 : active raid1 hda1[0] hdb1[1]
      979840 blocks [2/2] [UU]

unused devices: <none>
```

Pour faire le point sur les partitions `LVM` :

```
# lvs
  LV               VG     Attr   LSize   Origin Snap%  Move Log Copy%  Convert
  home             vgRoot -wi-ao 952.00M
  root             vgRoot -wi-ao   4.66G
  swap             vgRoot -wi-ao 952.00M
  tmp              vgRoot -wi-ao 952.00M
  var              vgRoot -wi-ao   4.66G
  var_lib_vservers vgRoot -wi-ao  61.49G
```

3.2.3 Utilitaires de base

```
apt-get install less vim apt-file
apt-file update
```

3.3 Répertoire de scripts

Créer un répertoire */root/bin*, et l'ajouter au `PATH` afin que ses scripts soient visibles parmi les commandes du système :

```
mkdir /root/bin
echo 'PATH="$PATH:/root/bin"' >> /root/.bashrc
source /root/.bashrc
```

Ne pas oublier d'y rendre les scripts exécutables, au fur et à mesure des ajouts :

```
chmod +x /root/bin/*
```

3.4 Fichier `hosts`

Pour la suite de ce document (et pour l'administration en général), il sera plaisant de pouvoir utiliser des noms de domaine plutôt que des adresses IP. Ainsi, il faudra veiller à maintenir le fichier */etc/hosts* régulièrement à jour. Il accueillera les différentes adresses des différents serveurs qui seront installés par la suite.

```
echo -e '\n\
192.168.0.200\tHOST\n\
192.168.0.201\tWWW\n\
192.168.0.202\tSWSSH\n\
192.168.0.203\tSVN\n\
192.168.0.204\tDB' >> /etc/hosts

# >> /var/lib/vservers/NOM/etc/hosts dans le cas des VServers (pour la suite)
```

Les équivalents noms de domaine ne seront utilisés que dans les cas où leur utilisation ne peut pas poser de problème. Ainsi, ils seront utilisés pour définir des règles `iptables` (puisque `netfilter` n'enregistrera que les IP — sauf pour les règles de `prerouting`) ou des connexions SSH mais pas pour la configuration du serveur web ou des serveurs SSH (qui planteront en cas de problème dans la résolution `DNS`), par exemple.

Ce fichier sera à recopier dans les différents futurs serveurs (selon le niveau de sécurité à respecter, ce type d'information n'étant pas anodin).

3.5 Firewall

3.5.1 Politique `iptables`

La politique de `netfilter` pour ses principales `tables` sera de ne pas laisser passer les paquets pour qui il n'y a pas une règle qui ait été définie qui les implique directement. Avant de définir les politiques restrictives qui couperaient immédiatement la session SSH, il faut définir quelques règles de base.

Autorisation de la connexion SSH qui sert à se connecter au serveur à distance :

```
iptables -A INPUT -p tcp -d HOST --dport 42 -j ACCEPT
```

Autorisation automatique des réponses des connexions persistantes (principales évolutions de `iptables` par rapport à son ancêtre `ipchains`) :

```
iptables -m state -A INPUT --state RELATED,ESTABLISHED -j ACCEPT
iptables -m state -A OUTPUT --state RELATED,ESTABLISHED -j ACCEPT
```

Politique par défaut pour tous les autres cas de figure (plus rien ne passe) :

```
iptables -P INPUT DROP
iptables -P OUTPUT DROP
iptables -P FORWARD DROP
```

Ajoutons la possibilité de faire du DNS :

```
iptables -A OUTPUT -p udp --dport 53 -j ACCEPT
iptables -A OUTPUT -p tcp --dport 53 -j ACCEPT
```

Autrefois il aurait été possible de restreindre le protocole à l'`UDP` pour ce type de règle. Malheureusement, la nouvelle norme DNS qui se met en place au niveau mondial et qui deviendra très vite exclusive fonctionne avec `TCP`. Ainsi, ce type de règle pourrait permettre à un intrus d'ouvrir une connexion SSH sur un serveur distant écoutant sur le port 53. Pour être rigoureux, il faudrait ajouter l'option `-d` avec l'adresse des serveurs DNS.

Enfin, afin de pouvoir utiliser `apt` pour la suite des installations, il faut autoriser les téléchargement sur les serveurs Debian. Pour plus de commodité, tous les accès web seront autorisés (depuis le système principale, comme depuis les serveurs qui seront hébergés) :

```
iptables -A OUTPUT -p tcp --dport 80 -j ACCEPT
```

Cette règle devra absolument être supprimée, une fois le serveur réellement en production :

```
iptables -D OUTPUT -p tcp --dport 80 -j ACCEPT
```

Pour visualiser les règles et politiques en cours :

- Voir les règles avec les noms de domaine et le type de service standard associé aux ports (correspondance avec */etc/services*) :

```
iptables -L
```

- Sans les correspondances :

```
iptables -nL
```

– Voir les règles de `prerouting` :

```
iptables -t nat -nL PREROUTING
```

3.5.2 Restauration des règles `iptables`

Afin que cette politique ainsi que les prochaines règles qui seront définies perdurent à chaque redémarrage du serveur, il est essentiel de maintenir un fichier de sauvegarde à jour et d'ajouter un service au système.

Sauvegarde des règles :

```
iptables-save > /etc/firewall-rules
```

Le fichier du service est disponible en annexe E.

Pour l'installer, le placer dans */etc/init.d/* puis exécuter :

```
chmod +x /etc/init.d/firewall
update-rc.d firewall defaults
```

3.6 Sauvegardes incrémentales journalières

Brancher le disque dur externe et l'ajouter dans le `fstab` afin qu'il soit monté automatiquement :

```
mkdir /media/data
echo '/dev/sda1 /media/data ext3 defaults 0 2' >> /etc/fstab
mount /media/data
```

Installer `rdiff-backup` :

```
apt-get install rdiff-backup
```

Copier le script disponible en annexe F dans le dossier */root/bin*, puis exécuter :

```
chmod +x /root/bin/backup
echo '0 3 * * * root /root/bin/backup 2>> /var/log/backups.log' > /etc/cron.d/backups
```

Tous les jours à trois heures du matin, `rdiff-backup` fera une sauvegarde incrémentale du système sur le disque dur externe, via le répertoire */media/data/rdiffBackup*.

3.7 `Fail2ban`

`Fail2ban` repère le nombre de tentatives de connexions infructueuses à SSH pour une même adresse IP en consultant les `logs`. Si une IP échoue plus de `n` fois, il crée une règle `iptables` pour lui interdire l'accès, rendant automatiquement infructueuse toute nouvelle tentative.

Installation :

```
apt-get install fail2ban
```

Configuration :

```
cp /etc/fail2ban/jail.{conf,local}
```

Dans */etc/fail2ban/jail.local*, au niveau de la section `[ssh]`, remplacer par :

```
[ssh]

enabled = true
port    = 42
filter  = sshd
logpath  = /var/log/auth.log
maxretry = 6

[ssh-vservers]

enabled = true
port    = 2280,443
filter  = sshd
logpath  = /var/lib/vservers/*/var/log/auth.log
maxretry = 6
```

La section `[ssh-vservers]` trouvera tout son sens dans la suite de ce document. Penser par la suite à y changer les ports en fonction des adaptations et des ajouts.

Relancer le service :

```
/etc/init.d/fail2ban restart
```

3.8 Les VServers

Le chapitre suivant sera consacré à ceux-ci. En attendant plus d'explications, il faut préparer le système à les accueillir. Le projet prévoit un patch du noyau ainsi qu'une collection d'outils à installer.

Installation du nouveau noyau et des outils VServers :

```
echo -e "deb http://ftp.debian.org/debian/ testing main" >> /etc/apt/sources.list

echo -e\
"Package: linux-image\nPin: release o=Debian,a=testing,l=Debian\nPin-Priority: 990\n\n"\
"Package: *\nPin: release o=Debian,a=stable,l=Debian\nPin-Priority: 700" > /etc/apt/preferences

apt-get update
apt-get install linux-image-2.6.32-vserver-686 util-vserver
```

Pour une étanchéité parfaite entre les VServers et éventuellement pour imposer des mesures de quotas par la suite, il est conseillé d'activer le `tagging xid` qui consiste à rajouter un drapeau sur chaque fichier ou processus lancé depuis un VServer. Ainsi, en cas d'escalade de l'arborescence, le pirate ne pourra ni voir ni toucher les fichiers qui ne portent pas le même drapeau et qui ne concernent donc pas le VServer depuis lequel il est connecté.

Exécuter :

```
sed 's/\(\/var\/lib\/vservers\/\?[ \t]*ext3[ \t]*defaults\)/\1,tag/' -i /etc/fstab
```

Pour information, voici comment *taguer* des fichiers a posteriori :

```
chxid -URx -c NOM_VSERVER DOSSIER
```

Redémarrer pour prendre en compte le nouveau noyau et remonter la partition :

```
reboot
```

4. Un serveur de serveurs

4.1 Préambule : la virtualisation par containers

La virtualisation est une technologie permettant de faire fonctionner plusieurs systèmes d'exploitation simultanément sur une même machine. La virtualisation par containers — par opposition à la virtualisation assistée par matériel ou la paravirtualisation — a la particularité de faire partager un noyau unique entre le système principal[1] et les différents systèmes hébergés[2]. L'accumulation des systèmes sur la machine ne nécessite donc pas une escalade des ressources nécessaires, mais restreint les systèmes hébergés aux distributions GNU/Linux, utilisant un noyau standard.

Le système GNU/Linux utilisé sera celui de Debian, aussi bien pour le système hôte que les systèmes hébergés. La projet utilisé pour virtualiser les systèmes est Linux-VServers[3].

4.2 Un container par service

Afin d'assurer l'étanchéité entre les services, un service sera équivalent à un container (donc un système GNU/Linux virtualisé). Les containers sont plus solides que des `chroot`s et assurent l'impossibilité technique de pouvoir atteindre les ressources liées aux autres services, ainsi que le système principal et le matériel. En cas d'intrusion, seules les données concernant le service infiltré seront mises en péril. Dans la mesure ou réussir à pénétrer le système hôte permettrait de compromettre l'intégralité des services et du matériel, il est impératif qu'aucun service ne tourne dessus, afin de réduire au maximum les risques de failles.

Le schéma 4.1 représente la manière dont un utilisateur va accéder aux différents services depuis Internet. Les adresses IP sont précisées pour pointer du doigt le passage de l'adresse publique aux adresses privées et le fait que les containers sont accessibles directement par le réseau sans — en apparence seulement — passer par la machine hôte. Trois services sont mis en scène afin d'illustrer leur accès, chacun de ces services ayant un port attribué. Ce schéma représente les cas les plus classiques, dénués de difficulté de conception. Cette façon de fonctionner est toutefois la base de toute la conception du serveur. Il y a donc quatre systèmes Debian installés, avec un système hôte pour trois systèmes virtualisés, et une seule machine. Le routeur en question est un routeur D-Link DIR-300, faisant lui-même office de pare-feu.

4.3 Netfilter, ce dealer

Netfilter est le pare-feu du noyau Linux, et s'utilise à travers ses outils `iptables`. Il servira en tant que pare-feu bien sûr, mais surtout en tant que *switcheur*. Ainsi, en fonction du port d'arrivé sur la machine, il *préroutera* les paquets vers l'interface réseau virtuelle qui est attribuée au container à qui il est destiné. Sa politique par défaut, pour ses principales tables est l'interception des paquets. Chaque cas d'accès possible au serveur, aux containers, ou entre les containers sera donc défini par des règles spécifiques.

Chaque création de VServer, et donc de service spécifique, entraînera l'ajout de nouvelles règles `iptables`.

[1] Le *host*, ou encore le *dom0* si on emprunte le jargon de Xen.
[2] Les *guests* ou encore les *domU*.
[3] http://linux-vservers.org

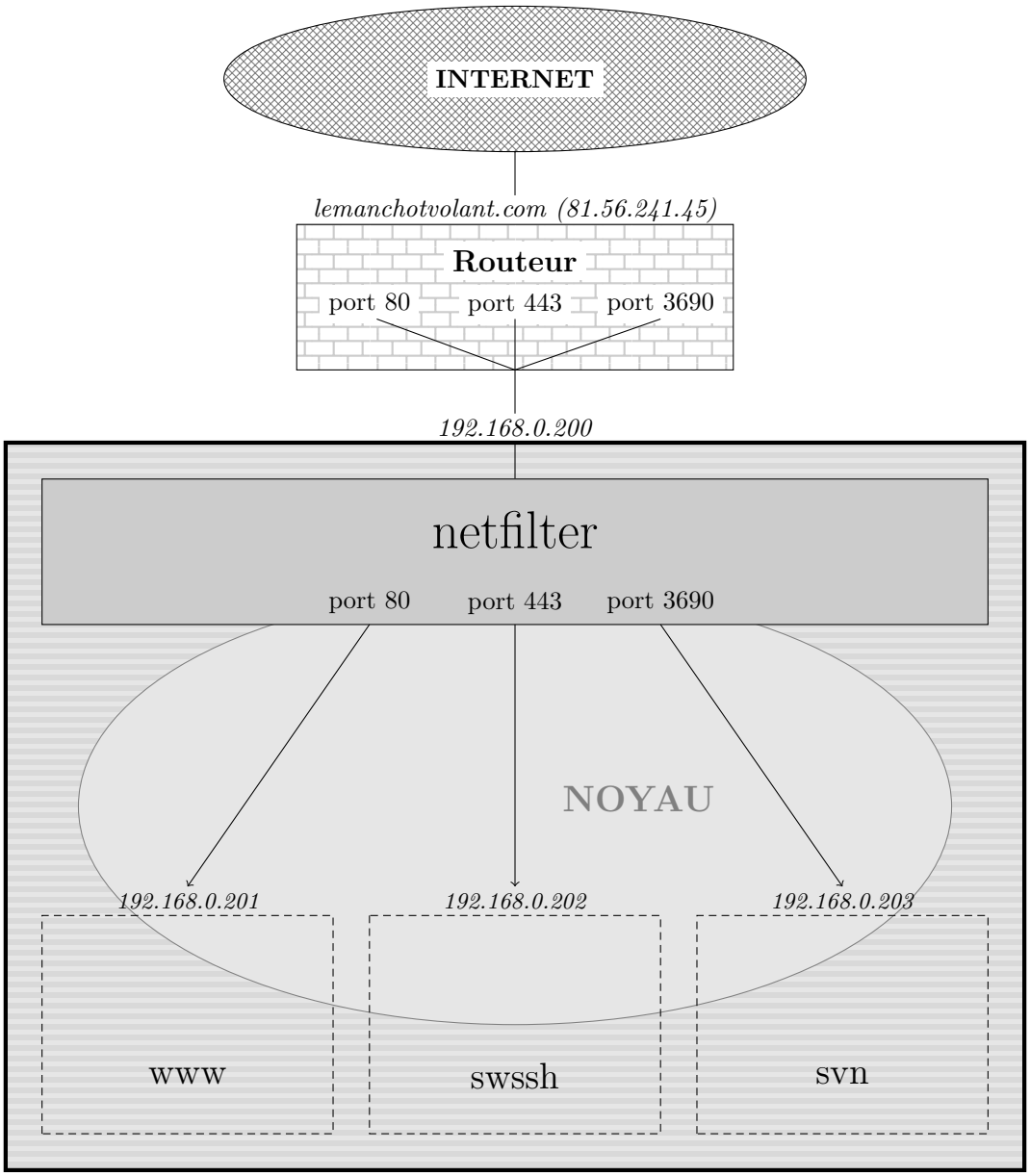


FIG. 4.1 – Organisation de l'accès aux services depuis Internet

4.4 Création d'un VServer

4.4.1 Partitionnement

Les outils de Linux-VServer permettent de limiter les ressources matérielles pour chaque container. Pour plus de sécurité encore, et pour rendre impossible l'immobilisation du serveur tout entier dans le cas où un container venait à se remplir de façon disproportionnée, il est intéressant de prévoir une partition par container. L'explication du choix initial du système de fichier LVM se révèle ici : à chaque création de container, la partition dédiée à ceux-ci sera fragmentée pour accueillir le nouveau système.

Pour cela, il faut :

1. Stopper tous les VServers déjà créés qui ont été lancés (`vserver NOM stop`)
2. Démonter toutes les partitions montées dans la partition des VServers (`umount /var/lib/vservers/NOM`)

3. Démonter la partition des VServers elle-même (`umount /var/lib/vservers`)
4. Lancer un *check* de cette partition, indispensable avant un éventuel redimensionnement (`e2fsck /dev/mapper/NOM_VG-NOM_LV`)
5. Redimensionner le système de fichier de la partition des VServers afin de libérer suffisamment d'espace pour la nouvelle partition (`resize2fs /dev/mapper/NOM_VG-NOM_LV XXG`)
6. Faire de même avec la partition elle-même (`lvresize -L XXG NOM_VG/NOM_LV`)
7. Créer la nouvelle partition (`lvcreate -L XXG -n NOM_LV`)
8. Créer son système de fichiers (`mkfs.ext3 /dev/mapper/NOM_VG-NOM_LV`)
9. Remonter la partition des VServers (`mount /var/lib/vservers`)
10. Créer le dossier du VServer (`mkdir /var/lib/vservers/NOM`)
11. Ajouter le point de montage dans le fichier `/etc/fstab`
12. Remonter toutes les partitions (`mount -a`)
13. Redémarrer tous les VServers (`vserver NOM start`)

Un script d'automatisation `vserverlv-new-partition` est disponible en annexe G (à ajouter dans le répertoire de scripts, cf. 3.3).

4.4.2 Installation

Une fois l'étape de partitionnement effectuée, un dossier */var/lib/vservers/nom* sur lequel une partition de la taille désirée est montée, est disponible.

Pour créer le VServer dedans :

```
vserver NOM build\
 -m debootstrap --context CTX\
 --hostname $nom.lemanchotvolant.com\
 --interface eth0:192.168.0.CTX/24\
 -- -d DEBIAN_VERSION -m http://ftp2.fr.debian.org/debian
```

NOM Nom du dossier qui a été créé précédemment
CTX Numéro de contexte du VServer : indiquer le dernier décimal de l'adresse IP qui lui sera attribué est une bonne solution
DEBIAN_VERSION `lenny`, `squeeze`, `sid`, etc.

Un script d'automatisation `vserverlv-new-install` est disponible en annexe H (à ajouter dans le répertoire de scripts, cf. 3.3).

4.4.3 Utilisation

Un VServer peut (doit !) être démarré (`start`), mais aussi arrêté (`stop`) ou relancé (`restart`). Son statut peut être consulté (`status`).

Pour entrer dans un VServer depuis le système principal (`Ctrl-D` pour le quitter) :

```
vserver NOM start
vserver NOM enter
```

Voir les VServers en cours d'exécution :

```
vserver-stat
```

Voir tous les processus réels de la machine (tous drapeaux confondus) :

```
vtop # ou
vps
```

En installant le paquet `vserver-debiantools`, il est possible d'administrer tous les VServers à la fois :

```
vapt-get --all -- update
vapt-get --all -- install locales
```

4.4.4 Lancement automatique

Par défaut, les VServer ne seront pas lancés au démarrage du système principal.

Pour forcer leur démarrage, exécuter :

```
echo default > /etc/vservers/NOM/apps/init/mark
```

Un script d'automatisation `vserver-autoboot` est disponible en annexe J, le placer dans le répertoire de scripts (cf. 3.3 pour le créer dans ce VServer).

4.4.5 Procédure complète

Un dernier script d'automatisation permet de lier toutes ces étapes : `vserverlv-new-all` (annexe **??**).

Exemple :

```
vserverlv-new-all -i 192.168.0.201 -n svn -s 5 -v lenny
```

Cet exemple, en plus de compléter la procédure avec l'installation des locales et la mise à jour du système, est l'équivalence des commandes suivantes :

```
ververlv-new-partition -n svn -s 5
vserverlv-new-install -n svn -v lenny -y
vserver-autoboot svn
vserver svn start
```

Aide :

```
vserverlv-new-all -h
```

Une aide pour chacun des scripts sous-jacent est accessible de la même façon.
Lancer ces scripts sans option offrira la possibilité de saisir les valeurs obligatoires interactivement.
Penser à compléter le fichier `hosts` du VServer (cf. 3.4).

4.4.6 Commandes supplémentaires

4.4.6.1 Renommer un VServer

La commande de l'exemple suivant se chargera de renommer le VServer ainsi que sa partition LVM (annexe K) :

```
vserverlv-rename ANCIEN_NOM NOUVEAU_NOM
```

Aide :

```
vserverlv-rename -h
```

4.4.6.2 Supprimer un VServer

La commande de l'exemple suivant se chargera de supprimer le VServer ainsi que sa partition LVM (annexe L) :

```
vserverlv-remove NOM
```

Aide :

```
vserverlv-remove -h
```

5. Serveur de BDD

5.1 Création du VServer

Il portera le nom de *db* et proposera une taille de *5*G (adapter les paramètres) :

```
ververlv-new-all -n db -i 192.168.0.204 -s 5 -v lenny
```

5.2 Installation d'un serveur MySQL

5.2.1 Installation classique

```
apt-get install mysql-server
```

Pour autoriser par la suite les connexions à distance, il faut commenter le `bind-address` du fichier */etc/mysql/my.cnf* :

```
sed '/bind-address[ \t]*= 127.0.0.1/ s/.*/#&/' -i /etc/mysql/my.cnf
```

5.2.2 Restaurer les bases de données

5.2.2.1 A partir de fichiers `dump`

```
mysql -u root -p < dump.sql
```

Le mot de passe MySQL `root` a été défini durant l'installation.

Si le fichier *dump.sql* ne contient pas d'instruction de création de la database concernée, il faut l'ajouter (ou la créer manuellement et utiliser `mysql -u root lemanchotvolant -p`) :

```
sed 1i"CREATE DATABASE lemanchotvolant;" -i dump.sql
mysql -u root -p < dump.sql
```

5.2.2.2 A partir du répertoire */var/lib/mysql*

Si le cadre de cette lecture est une réinstallation de serveur, alors il est possible d'utiliser directement les données d'un répertoire */var/lib/mysql*. Commencer par l'envoyer sur le répertoire personnel de `auk`.

Puis, depuis le système principal :

```
rm -r /var/lib/vservers/db/var/lib/mysql
mv /home/auk/mysql /var/lib/vservers/db/var/lib
```

Et depuis le VServer :

```
chown -R mysql:mysql /var/lib/mysql/*
chown root:root /var/lib/mysql/debian-5.0.flag,mysql_upgrade_info
/etc/init.d/mysql restart
```

Une erreur devrait se manifester :

```
/usr/bin/mysqladmin: connect to server at 'localhost' failed
error: 'Access denied for user 'debian-sys-maint'@'localhost' (using password: YES)'
```

L'utilisateur `debian-sys-maint` est utilisé par MySQL pour les opérations de maintenance. Son mot de passe est généré aléatoirement à l'installation. Ayant écrasé ses informations avec celles de l'ancien serveur, il est donc logique que le mot de passe ne corresponde plus. Malgré l'erreur, le serveur est lancé, en prenant en compte les nouvelles (anciennes) bases, tables, ainsi que les utilisateurs. Il faut récupérer le mot de passe actuel de `debian-sys-maint` et le mettre à jour dans la base de données.

Une solution radicale consiste à écraser le fichier */etc/mysql/debian.cnf* avec celui de l'ancien serveur. Sinon, mettre à jour le mot de passe comme ceci :

```
echo 'GRANT ALL PRIVILEGES ON *.* TO debian-sys-maint@localhost IDENTIFIED BY\
 '$(grep password /etc/mysql/debian.cnf | head -n1 | cut -d" " -f3)'\
 WITH GRANT OPTION;' >> /tmp/req

mysql -u root -p < /tmp/req && rm /tmp/req
```

5.3 Gestion des utilisateurs

Il est de bon ton de créer un utilisateur rattaché à une base spécifique et ayant éventuellement des droits limités, pour chacun des sites créés. Les utilisateurs seront définis avec un hôte spécifique, évitant ainsi les accès depuis d'autres serveurs qui ne seraient pas légitimes.

Créer l'utilisateur depuis le client MySQL (`mysql -u root -p`) :

```
GRANT INSERT,UPDATE,DELETE ON lemanchotvolant.* TO manchot@WWW IDENTIFIED BY 'PASSWORD';
```

L'utilisateur `manchot` aura les droits d'insertion, de mise à jour et de suppression d'entrées dans toutes les tables de la base `lemanchotvolant`. `WWW` correspond au nom de domaine reliantle VServer qui hébergera le serveur web.

Attention, le fichier `$HOME/.mysql_history` qui contient toutes les commandes MySQL, avec éventuellement les mots de passe en clair des utilisateurs créés, est automatiquement écrit.

5.4 Utilisateurs et SSH

Lorsqu'un VServer est créé, aucun utilisateur n'est créé et le compte `root` n'a pas de mot de passe (équivalent à `passwd root -d`). Dans la mesure où il n'a a aucune raison de se connecter directement sur ce serveur, sauf pour l'administrer, cette configuration convient. L'absence de comptes et l'absence de SSH restreint l'accès au serveur à la commande `vserver` du système principal.

5.5 Règles `iptables`

Sur le système principal :

```
iptables -A OUTPUT -p tcp -s WWW -d DB --dport 3306
iptables -A INPUT -p tcp -s WWW -d DB --dport 3306
```

Seuls les échanges entre le serveur web (`WWW`) et le serveur de base données (`DB`) seront possibles. La règle `OUTPUT` sert lorsque le paquet « sort » du VServer du serveur web, tandis que la règle `INPUT` sert lorsque ce même paquet rentre dans le VServer de base de données.

Ne pas oublier de mettre à jour le fichier de sauvegarde (cf. 3.5.2).

6. Serveur web

6.1 Création du VServer

Il portera le nom de *www* et proposera une taille de *13*G (adapter les paramètres) :

```
ververlv-new-all -n www -i 192.168.0.201 -s 13 -v testing
```

6.2 Installation des logiciels

6.2.1 Installation du serveur

```
apt-get install apache2 php5 php5-mysql libapache2-mod-php5 mysql-client
```

6.2.2 Installation de PDO

```
apt-get install php-db php5-dev libmysqlclient15-dev
```

php-db Installe `pear` et `php5-cli`
php5-dev Installe `phpize`
libmysqlclient15-dev Sinon, erreur `Cannot find MySQL header files` par la suite

Exécuter (force la regénération, sinon erreur `pecl.php.net is using a unsupported protocal`) :

```
rm -rf /usr/share/php/.channels
```

Installation :

```
pear update-channels
pecl install pdo
pecl install pdo_mysql
```

Ajouter dans */etc/php5/apache2/php.ini* :

```
extension=pdo.so
extension=pdo_mysql.so
```

6.3 Utilisateurs

L'utilisateur `www` sera l'utilisateur principal. Son home sera directement */var/www*.

```
adduser www --home /var/www --no-create-home --ingroup www-data
chown -R www:www-data /var/www
```

Un utilisateur `root` peut être utile pour les opération de maintenance du serveur :

```
passwd
```

6.4 Configuration DNS

Les DNS du Manchot volant sont gérés par OVH.

Ainsi, dans le *Manager*, une fois le NDD sélectionné, dans *Domaine & DNS* puis dans *Zone DNS*, on trouve :

.lemanchotvolant.com	NS	dns15.ovh.net
.lemanchotvolant.com	NS	ns15.ovh.net
.lemanchotvolant.com	MX 1	redirect.ovh.net
.lemanchotvolant.com	A	81.56.241.45
*.lemanchotvolant.com	CNAME	lemanchotvolant.com

Les deux `NS` ne peuvent être changés, en l'absence de serveur mail le `MX` est géré par OVH et l'adresse IP de la connexion Internet est renseignée en tant que correspondance. Enfin, tous les sous-domaines sont renvoyés vers le domaine `www`, ainsi toute la gestion se fera selon la configuration du serveur destinataire.

6.5 Gestion des sites

Les sites sont à la racine de */var/www*, et leur nom de répertoire sont préfixés de `site-`.

Les sites ne sont gérés que par `VirtualHost`. Le fichier */etc/apache2/ports.conf* est configuré comme ceci :

```
Listen 80
NameVirtualHost 192.168.0.201
```

Supprimer la configuration du `VirtualHost` par défaut :

```
rm /etc/apache2/site-available,enabled/*
```

Un répertoire */var/www/site-defaut* sera créé pour faire aboutir dans un trou noir toutes les requêtes non prévues par la configuration :

```
mkdir /var/www/site-default
chmod 0000 /var/www/site-default
```

Fichier */etc/apache2/sites-available/defaut* :

```
<VirtualHost 192.168.0.201>
        DocumentRoot /var/www/site-default

        <Directory /var/www/site-default>
                Order deny,allow
                Allow from all
        </Directory>
</VirtualHost>
```

Site principal (*/etc/apache2/site-available/lemanchotvolant*) :

```
<VirtualHost 192.168.0.201>
        UseCanonicalName Off
        ServerName www.lemanchotvolant.com
        DocumentRoot /var/www/site-lemanchotvolant

        <Directory /var/www/site-lemanchotvolant>
                Order allow,deny
                allow from all
        </Directory>
</VirtualHost>
```

Ses fichiers seront donc stockés dans */var/www/site-lemanchotvolant*.

Exemple de sous-domaine (*/etc/apache2/site-available/wiki*) :

```
<VirtualHost 192.168.0.201>
        UseCanonicalName Off
        ServerName wiki.lemanchotvolant.com
        DocumentRoot /var/www/site-wiki

        <Directory /var/www/site-wiki>
                Order allow,deny
                allow from all
        </Directory>
</VirtualHost>
```

Ses fichiers seront donc stockés dans */var/www/site-wiki*.

Activez chacun de ces sites (ex : `a2ensite wiki`) en commencant par *defaut* et assurez-vous que les liens symboliques dans */etc/apache2/site-enabled* sont préfixés d'un nombre à trois chiffres et que *defaut* est le plus petit :

```
www:/etc/apache2/sites-enabled# ls
000-defaut  001-lemanchotvolant  002-wiki
```

6.6 Sécuriser un site par mot de passe

Exemple pour le sous-domaine (*/etc/apache2/site-available/wiki*) :

```
<VirtualHost 192.168.0.201>
        UseCanonicalName Off
        ServerName wiki.lemanchotvolant.com
        DocumentRoot /var/www/site-wiki

        <Directory /var/www/site-wiki>
                Order allow,deny
                allow from all

                AuthName "Wiki"
                AuthType Digest
                AuthDigestProvider file
                AuthUserFile /var/www/digest/.lmv
                Require valid-user
        </Directory>
</VirtualHost>
```

Création du fichier de mot de passe :

```
a2enmod auth_digest
mkdir /var/www/digest
htdigest -c /var/www/digest/.wiki Wiki ju
```

Avec l'option `AllowOverride All` il est possible de de stocker les directives `Auth*` et `Require` dans un fichier *.htaccess*, plus commode à manipuler puisqu'il ne nécessite pas de relancer le serveur web.

6.7 Sécurisation du serveur web

Dans */etc/apache2/conf.d/security*, changer quelques paramètres permettant de cacher quelques informations sur le serveur, trop facilement accessibles :

```
sed -e 's/^ServerTokens [a-z]*/ServerTokens Prod/i'  -e 's/^ServerSignature [a-z]*/ServerSignature Off/i'
-e 's/^TraceEnable [a-z]*/TraceEnable Off/i'  -i /etc/apache2/conf.d/security
```

6.8 Accès SSH

Installation de SSH :

```
apt-get install ssh
```

Points remarquables de la configuration (*/etc/ssh/sshd.config*) :

```
Port 2280
ListenAddress 192.168.0.201

PermitRootLogin no
PasswordAuthentication no
AuthorizedKeysFile %h/.ssh/authorized_keys

AllowUser www
```

Le port SSH sera le port 2280, et l'accès ne sera possible que pour l'utilisateur www, avec une clé SSH.
Le fichier complet est disponible en annexe M.

Relancer le serveur :

```
/etc/init.d/ssh restart
```

6.9 Règles `iptables`

Tout accès qui arrive sur le serveur par le port 80 concerne le serveur web, ainsi on *préroute* les paquets directement vers le VServer concerné. Idem pour le port 2280 utilisé pour le SSH du VServer.

Depuis le système principal :

```
iptables -t nat -A PREROUTING -p tcp --dport 80 -j DNAT --to 192.168.0.201
iptables -t nat -A PREROUTING -p tcp --dport 2280 -j DNAT --to 192.168.0.201
```

Bien que les paquets soient redirigés directement vers le VServer concerné, ils passent par le système principale.

Ainsi, il faut aussi autoriser leur passage :

```
iptables -A INPUT -p tcp -d WWW --dport 80 -j ACCEPT
iptables -A INPUT -p tcp -d WWW --dport 2280 -j ACCEPT
```

Ne pas oublier de mettre à jour le fichier de sauvegarde (cf. 3.5.2).

7. Serveur SVN

7.1 Création du VServer

Il portera le nom de *svn* et proposera une taille de *5*G (adapter les paramètres) :

```
ververlv-new-all -n svn -i 192.168.0.203 -s 5 -v lenny
```

7.2 Installation du serveur

7.2.1 Paquet Debian

```
apt-get install subversion
```

7.2.2 Service SVN

Curieusement, le paquet Debian n'installe pas de service pour lancer le serveur SVN.
Un script est disponible en annexe N afin de combler ce manque.

Copier ce fichier dans */etc/init.d/* puis le rendre exécutable :

```
chmod +x /etc/init.d/svnserve
```

7.3 Principe de gestion

La gestion classique des utilisateurs d'un projet SVN se fait généralement dans le fichier */var/svn/projet/conf/passwd*. Les mots de passe sont en clair, les comptes dupliqués à chaque projet et tout transite sans cryptage. Autant dire qu'il est impossible d'envisager cette solution. L'astuce est donc d'utiliser le protocole SSH et les utilisateurs et groupes Unix du système.

Chaque nouveau projet donnera lieu à la création d'un nouveau groupe Unix du même nom. Chaque utilisateur rattaché au projet sera ajouté à ce groupe, et éventuellement créé pour l'occasion. Les connexions au projet SVN se feront par le protocole `svn+ssh`, et utiliseront une clé SSH. L'accès SSH sera limité à la seule tâche de la communication avec le serveur SVN.

7.4 Installer SSH

```
apt-get install ssh
```

Points remarquables de la configuration (*/etc/ssh/sshd_config*) :

```
Port 3690
ListenAddress 192.168.0.203

PermitRootLogin no
PasswordAuthentication no
AuthorizedKeysFile %h/.ssh/authorized_keys
```

Le port SSH sera le port 3690 (le port habituellement utilisé par SVN), et l'accès ne sera possible qu'avec une clé SSH.
Le fichier complet est disponible en annexe O.

Relancer le serveur :

```
/etc/init.d/ssh restart
```

7.5 Gestion de projet

7.5.1 Répertoires utilisateurs

Afin de créer correctement les `home` des utilisateurs Unix de façon automatisée, il faut adapter le contenu du répertoire */etc/skel*.
Commencer par y supprimer l'intégralité des fichiers, dont ceux cachés.

Puis, simplement créer le dossier *.ssh* et le fichier des clés publiques :

```
mkdir /etc/skel/.ssh
echo 'command="/usr/bin/svnserve -t -r /var/svn" ' > /etc/skel/.ssh/authorized_keys
```

La ligne qui pré-remplie le fichier des clés restreint l'utilisation du SSH à sa seule fonction de relais vers le serveur SVN. L'option `-t` indique qu'on souhaite lancer une session SVN qu'on utilisera à travers un tunnel, et l'option `-r` permet de changer la racine des projets et ainsi simplifier par la suite l'accès aux répertoires (et éventuellement accroître la sécurité en en disant un peu moins sur l'arborescence du serveur).

7.5.2 Créer un projet

Pour créer un projet SVN :

1. Créer le groupe Unix associé (`addgroup -system` *PROJET*)
2. Créer le projet (`svnadmin create /var/svn/`*PROJET*)
3. Attribuer les fichiers du projet au groupe associé (`chgrp -R` *PROJET* `/var/svn/`*PROJET*)
4. Limiter les droits de l'ensemble du projet (`chmod -R 0660 /var/svn/`*PROJET*)
5. Rendre le droit d'exécution aux dossiers (`find /var/svn/`*PROJET* `-type d -exec chmod ug+x {} \+`)
6. Créer les utilisateurs associés au projet (`useradd -m -k /etc/skel -g` *PROJET NOM*) ou simplement les ajouter au groupe si ils existent déjà (`adduser` *NOM PROJET*)
7. Ajouter la clé publique des utilisateurs dans le fichier *.ssh/authorized_keys* correspondant à leur `home`, en prenant garde de ne pas supprimer la ligne déjà présente (il faut ajouter la clé juste après, sur la même ligne, avec juste un espace)

7.5.3 Script

Un script d'automatisation `svn-add` pour la création de projet ou l'ajout d'utilisateur à un projet est disponible en annexe P.
Au lancement de celui-ci, deux choses seront à renseigner : un nom de projet et un nom d'utilisateur.

- Quand le projet n'existe pas, il le crée correctement (ainsi que le groupe), et ajoute l'utilisateur au projet (ainsi qu'au groupe)
- Quand l'utilisateur n'existe pas, il le crée et l'ajoute au groupe du projet
- Quand l'utilisateur existe déjà et le projet aussi, il se contente d'ajouter l'utilisateur au groupe

Ce script sert donc dans tous les cas, l'ajouter au répertoire de scripts (cf. 3.3 pour le créer dans le VServer).

7.6 Utilisateur `root`

Inutile de disposer d'un utilisateur accessible, la seule possibilité d'administration sera donc la commande `vserver` du système principal, rendant ainsi impossible la prise en main complète du serveur encas d'attaque (équivalent de `passwd root -d`).

7.7 Règles `iptables`

Depuis le système principal :

```
iptables -t nat -A PREROUTING -p tcp --dport 3690 -j DNAT --to 192.168.0.203
iptables -A INPUT -p tcp -d SVN --dport 3690 -j ACCEPT
```

Ne pas oublier de mettre à jour le fichier de sauvegarde (cf. 3.5.2).

8. Serveur de passerelle SSH

8.1 Création du VServer

Il portera le nom de *swssh* (pour *Switch SSH*) et proposera une taille de *1*G (adapter les paramètres) :

```
ververlv-new-all -n svn -i 192.168.0.202 -s 1 -v lenny
```

8.2 Principe

Le VServer `swssh` (pour *switch SSH*) aura deux utilités :
– Le rebond SSH, pour outrepasser les proxies
– Le tunneling SSH, pour outrepasser les proxies ou simplifier des transferts
Il disposera de comptes utilisateurs, qui auront accès à leur espace personnel. Il est donc indispensable de verrouiller au maximum les possibilités offertes par cet espace.

8.3 Installation de SSH

Installation de SSH :

```
apt-get install ssh
```

Points remarquables de la configuration (*/etc/ssh/sshd_config*) :

```
Port 443
ListenAddress 192.168.0.202

PermitRootLogin no
PasswordAuthentication no
AuthorizedKeysFile %h/.ssh/authorized_keys

AllowTcpForwarding yes
```

Le port SSH sera le port 443 (logique, puisque le but est de pouvoir atteindre ce serveur à travers un proxy), et l'accès ne sera possible qu'avec une clé RSA. On peut remarquer l'autorisation du *tunneling* SSH.
Le fichier complet est disponible en annexe Q.

8.4 Sécurisation

8.4.1 Désarmement

On prendra soin de rendre inexécutable quelques binaires qui ne sont pas nécessaires aux utilisateurs et qui pourraient être utilisés en cas de tentative d'attaque :

```
chmod o-x /bin/chown /bin/su /bin/kill /bin/mount /bin/umount /bin/dmesg /bin/chmod /usr/bin/wget
```

8.4.2 Droits des fichiers des utilisateurs

Supprimer tous les fichiers sauf le *.bash_profile* du répertoire */etc/skel*.

Créer le fichier des clés publiques SSH dans le répertoire squelette des `home` :

```
mkdir /etc/skel/.ssh
touch /etc/skel/.ssh/authorized_keys
```

Pour créer un nouvel utilisateur :

1. Créer l'utilisateur Unix (`useradd -m -k /etc/skel` *NOM*)
2. Fermer les droits de son `home` (`chmod 0700 /home/`*NOM*)
3. Rendre à `root` le fichier des clées publiques SSH, afin que l'utilisateur ne puisse pas en rajouter (`chown root :root /home/`*NOM*`/.ssh/authorized_keys`)
4. Donner les bons droits à ce fichier (`chmod 644 /home/`*NOM*`/.ssh/authorized_keys`)

Un script d'automatisation `user-add` est disponible en annexe R, le placer dans le répertoire de scripts (cf. 3.3 pour le créer dans ce VServer).

8.5 Utilisateur `root`

Interdiction de se connecter en `root` au serveur, et impossible de le devenir (plus de `su` ni autres). La seule façon d'administrer le serveur est la commande `vserver` du système principal.

8.6 Règles `iptables`

Depuis le système principal :

```
iptables -t nat -A PREROUTING -p tcp --dport 443 -j ACCEPT --to 192.168.0.202
iptables -A INPUT -p tcp -d SWSSH --dport 443 -j ACCEPT
iptables -A OUTPUT -p tcp -s SWSSH -d HOST --dport 42 -j ACCEPT
iptables -A OUTPUT -p tcp -s SWSSH -d WWW --dport 2280 -j ACCEPT
iptables -A OUTPUT -p tcp -s SWSSH --dport 22 -j ACCEPT
```

Ne pas oublier de mettre à jour le fichier de sauvegarde (cf. 3.5.2).

Sur ce VServer, dans l'ordre, on peut donc :
- Arriver en 443 dessus
- Se connecter en SSH (42) sur le système principal (supprimer cette règle renforcerait la sécurité)
- Se connecter en SSH (2280) au serveur web
- Se connecter en SSH (22) n'importe où (il serait largement préférable de plutôt créer une règle `iptables` par serveur connu)

9. Références

En vrac quelques liens :

RAID 1 http://ubuntu.le7.net/2008/05/installation-debian-ubuntu-avec-raid-1
VServers http://linux-vserver.org
VServers http://www.atolcd.com/uploads/media/vserver_atol_web.pdf
Sécurité http://www.alsacreations.com/tuto/lire/622-Securite-firewall-iptables.html
Bash 3 http://fr.opensuse.org/Bash
Bash 3 http://abs.traduc.org/abs-5.0-fr/ch09s02.html

10. Evolutions

- Centralisation LDAP de comptes
- Serveur APT interne
- Serveur Postfix / Webmail
- Proxy
- Rkhunter (anti-rootkit)
- Ip over dns
- Serveur jabber

A. Matériel (extérieur)

FIG. A.1 – Le serveur du Manchot volant

B. Matériel (intérieur)

FIG. B.1 – Les entrailles du Manchot

C. Service dhcp

```
#!/bin/sh
## Author <julien@vaubourg.com>

### BEGIN INIT INFO
# Provides:             ssh
# Required-Start:
# Required-Stop:
# Default-Start:        2 3 4 5
# Default-Stop:         1
# Short-Description:    Launch a dhclient
### END INIT INFO

case "$1" in
        start)
                dhclient eth0
        ;;

        restart)
                /etc/init.d/dhcp start
        ;;

        *)
                echo "Usage: /etc/init.d/dhcp (re)start"
                exit 1
esac

exit 0
```

FIG. C.1 – [Système principal], fichier ***__HOST/etc/init.d/dhcp***

D. Config. `sshd_config`

```
Port 42
ListenAddress 192.168.0.200
Protocol 2

HostKey /etc/ssh/ssh_host_rsa_key
HostKey /etc/ssh/ssh_host_dsa_key

UsePrivilegeSeparation yes
KeyRegenerationInterval 3600
ServerKeyBits 768
SyslogFacility AUTH
LogLevel INFO

LoginGraceTime 120
PermitRootLogin no
StrictModes yes

RSAAuthentication yes
PubkeyAuthentication yes
AuthorizedKeysFile %h/.ssh/authorized_keys

IgnoreRhosts yes
RhostsRSAAuthentication no
HostbasedAuthentication no
PermitEmptyPasswords no

ChallengeResponseAuthentication no
PasswordAuthentication no

X11Forwarding no
X11DisplayOffset 10
PrintMotd no
PrintLastLog yes
TCPKeepAlive yes

AcceptEnv LANG LC_*
Subsystem sftp /usr/lib/openssh/sftp-server

UsePAM yes
AllowUsers auk
```

FIG. D.1 – [Système principal], fichier ***__HOST/etc/ssh/sshd_config***

E. Service `firewall`

```
#!/bin/sh
## Author <julien@vaubourg.com>

### BEGIN INIT INFO
# Provides:             iptables-restore
# Required-Start:
# Required-Stop:
# Default-Start:        2 3 4 5
# Default-Stop:         1
# Short-Description:    Restore iptables rules
### END INIT INFO

case "$1" in
        start)
                iptables-restore < /etc/firewall-rules
        ;;

        restart)
                /etc/init.d/firewall start
        ;;

        *)
                echo "Usage: /etc/init.d/firewall (re)start"
                exit 1
esac

exit 0
```

FIG. E.1 – **[Système principal]**, fichier ***__HOST/etc/init.d/firewall***

F. Script `backup`

```
#!/bin/sh
## Author <julien@vaubourg.com>

rdiff-backup --exclude-regexp '/(proc|sys|dev|tmp/*|media/*|mnt/*|/var/tmp/*|/var/lib/vservers/*/media/*)' / /media/data/rdiffBackup

exit 0
```

FIG. F.1 – **[Système principal]**, fichier ***__HOST/root/bin/backup***

G. Script vserverlv-new-partition

```
#!/bin/bash
## Author <julien@vaubourg.com>

if [ $UID -ne 0 ]; then
        echo -e "ERREUR:\n\tVous devez avoir les droits root" >&2
        exit 1
fi

if [ -z "$(which vserver)" ]; then
        echo -e "ERREUR:\n\tLes outils VServer ne semblent pas disponibles.\n\tVerifiez que vous etes bien sur le systeme principal et que le PATH est
        exit 1
fi

help() {
        echo -e "\n\tRedimensionne la partition des VServers (var_lib_vservers) pour en creer une nouvelle (var_lib_vservers_<nom>)."
        echo -e "\tA l'issu du script, un dossier /var/lib/vservers/<nom> est cree, point de montage de la nouvelle partition de <taille>G,\n\tpret a a

        echo -e "\n\tAttention : ce script arrete tous les VServers et demonte leurs partitions le temps de son execution."

        echo -e "\n\tOPTIONS"
        echo -e "\t\t-h : Affiche la presente aide"
        echo -e "\t\t-n : <nom> de la partition (qui sera par la suite le nom du VServer)"
        echo -e "\t\t-s : <taille> de la nouvelle partition en G (la partititons des VServers sera reduite d'autant)"
        echo -e "\t\t-f : Autorise automatiquement chacune des etapes du script"

        echo -e "\n\tNOTE : Si une option obligatoire n'est pas precisee, elle sera demandee interactivement."

        echo -e "\n\tEXEMPLES"
        echo -e "\t\t./vserverlv-new-partition -n toto -s 5 -y"
        echo -e "\t\t./vserverlv-new-partition"
        echo -e "\t\t./vserverlv-new-partition -yn toto"

        echo -e "\n\tAUTEUR"
        echo -e "\t\tJulien VAUBOURG <julien@vaubourg.com>\n"
}

while getopts hfn:s: option
do
        case $option in
                h)
                        help
                        exit 0
                ;;

                n)
                        lvName=$OPTARG
                ;;

                s)
                        tailleNew=$OPTARG
                ;;

                f)
                        forceYes=1
                ;;

                *)
                        help
                        exit 1
        esac
done

## Nom du groupe de lv
vg=vgRoot

## Chemin des VServers, qui servira pour construire le nom de la partition
lvPath=/var/lib/vservers

action() { ((i++))

        if [ -z "$forceYes" ]; then
                echo -e "\n($i/${#cmd[*]}) $1"
                echo -n "Executer : <$2> (Y/n) "
                read execmd
        fi

        if [ -z "$execmd" -o "$execmd" == "y" -o "$execmd" == "Y" ]; then
                [ -z "$forceYes" ] || echo -e "\n### $1..."

                eval "$2"
        else
                echo "NON"
        fi
}

## Supprime le slash en prefixe
```

H. Script vserverlv-new-install

```
#!/bin/sh
## Author <julien@vaubourg.com>

if [ $UID -ne 0 ]; then
        echo -e "ERREUR:\n\tVous devez avoir les droits root" >&2
        exit 1
fi

help() {
        echo -e "\n\tTelecharge les sources de Debian <version> pour installer un VServer <nom> dans le dossier /var/lib/vservers/<nom>."
        echo -e "\tL'IP a attribuer doit etre indiquee, et son dernier nombre servira de contexte pour le VServer."

        echo -e "\n\tOPTIONS"
        echo -e "\t\t-h : Affiche la presente aide"
        echo -e "\t\t-i : <IP> du nouveau VServer (le dernier chiffre sert de contexte au VServer)"
        echo -e "\t\t-n : <nom> du VServer (qui s'installera dans /var/lib/vservers/<nom>)"
        echo -e "\t\t-v : <version> de Debian a installer (lenny, squeeze, sid, etc.)"

        echo -e "\n\tNOTE : Si une option obligatoire n'est pas precisee, elle sera demandee interactivement."

        echo -e "\n\tEXEMPLES"
        echo -e "\t\t./vserverlv-new-install -n toto -i 192.168.0.201 -v lenny"
        echo -e "\t\t./vserverlv-new-install"
        echo -e "\t\t./vserverlv-new-install -n toto"

        echo -e "\n\tAUTEUR"
        echo -e "\t\tJulien VAUBOURG <julien@vaubourg.com>\n"
}

while getopts hn:i:v: option
do
        case $option in
                h)
                        help
                        exit 0
                ;;

                n)
                        nom=$OPTARG
                ;;

                i)
                        ip=$OPTARG
                ;;

                v)
                        version=$OPTARG
                ;;

                *)
                        help
                        exit 1
        esac
done

if [ -z "$nom" ]; then
        echo -n "Nom du VServer : "
        read nom
fi

if [ -z "$ip" ]; then
        echo -n "IP : "
        read ip
fi

ctx=${ip##*.}

if [ -z "$version" ]; then
        echo -n "Version de Debian : "
        read version
fi

vserver $nom build \
        -m debootstrap --context $ctx \
        --hostname $nom.lemanchotvolant.com \
        --interface eth0:$ip/24 \
        -- -d $version -m http://ftp2.fr.debian.org/debian
```

FIG. H.1 – [Système principal], fichier ***__HOST/root/bin/vserverlv-new-install***

I. Script vserverlv-new-all

```
#!/bin/bash

help() {
        echo -e "\n\t1) Redimensionne la partition des VServers (var_lib_vservers) pour en creer une nouvelle (var_lib_vservers_<nom>)."
        echo -e "\tA l'issu du script, un dossier /var/lib/vservers/<nom> est cree, point de montage de la nouvelle partition de <taille>G,\n\tpret a a

        echo -e "\n\t2) Telecharge les sources de Debian <version> pour installer un VServer <nom> dans le dossier /var/lib/vservers/<nom>."
        echo -e "\tL'IP a attribuer doit etre indiquee, et son dernier nombre servira de contexte pour le VServer."

        echo -e "\n\tAttention : ce script arrete tous les VServers et demonte leurs partitions le temps de son execution."

        echo -e "\n\tCe script est la compilation des scripts suivants :"
        echo -e "\t\tvserverlv-new-partition (aide : -h)"
        echo -e "\t\tvserverlv-new-install (aide : -h)"
        echo -e "\t\tvserver-autoboot (aide : -h)"

        echo -e "\n\t\tPlus l'installation de SSH et des locales."

        echo -e "\n\tOPTIONS"
        echo -e "\t\t-h : Affiche la presente aide"
        echo -e "\t\t-i : <IP> du nouveau VServer (le dernier chiffre sert de contexte au VServer)"
        echo -e "\t\t-n : <nom> du VServer (qui s'installera dans /var/lib/vservers/<nom>)"
        echo -e "\t\t-s : <taille> de la nouvelle partition dy VServer en G (la partititons des VServers sera reduite d'autant)"
        echo -e "\t\t-v : <version> de Debian a installer (lenny, squeeze, sid, etc.)"

        echo -e "\n\tNOTE : Si une option obligatoire n'est pas precisee, elle sera demandee interactivement."

        echo -e "\n\tEXEMPLES"
        echo -e "\t\t./vserverlv-new-all -n toto -s 2 -i 192.168.0.201 -v lenny"
        echo -e "\t\t./vserverlv-new-all"
        echo -e "\t\t./vserverlv-new-all -n toto"

        echo -e "\n\tAUTEUR"
        echo -e "\t\tJulien VAUBOURG <julien@vaubourg.com>\n"
}

while getopts hi:n:v:s: option
do
        case $option in
                h)
                        help
                        exit 0
                ;;

                i)
                        VSNewArgs="$VSNewArgs -i $OPTARG"
                ;;

                n)
                        VSlvNewArgs="$VSlvNewArgs -n $OPTARG"
                        VSNewArgs="$VSNewArgs -n $OPTARG"
                        nom=$OPTARG
                ;;

                s)
                        VSlvNewArgs="$VSlvNewArgs -s $OPTARG"
                ;;

                v)
                        VSNewArgs="$VSNewArgs -v $OPTARG"
                ;;


                *)
                        help
                        exit 1
        esac
done

echo -e "\n######"
echo "###### PARTITIONNEMENT"
echo -e "######\n"

vserverlv-new-partition $VSlvNewArgs -f

echo -e "\n######"
echo "###### INSTALLATION"
echo -e "######\n"

vserverlv-new-install $VSNewArgs
vserver-autoboot $nom

vserver $nom start
vserver $nom apt-get update
vserver $nom apt-get install ssh locales --force-yes
```

J. Script vserver-autoboot

```
#!/bin/sh
## Author <julien@vaubourg.com>

if [ $UID -ne 0 ]; then
        echo -e "ERREUR:\n\tVous devez avoir les droits root" >&2
        exit 1
fi

if [ $# -lt 1 -o "$1" == "-h" ]; then
        echo -e "\n\tActive le lancement automatique de VServers au demarrage du systeme principal."

        echo -e "\n\tUSAGE"
        echo -e "\t\tvserver-autoboot vserver [vserver ...]"

        echo -e "\n\tAUTEUR"
        echo -e "\t\tJulien VAUBOURG <julien@vaubourg.com>\n"

        [ $# -lt 1 ] && exit 1 || exit 0
fi

for i in $*; do
        echo default > /etc/vservers/$i/apps/init/mark
done

exit 0
```

FIG. J.1 – **[Système principal]**, fichier ***__HOST/root/bin/vserver-autoboot***

K. Script vserverlv-rename

```
#!/bin/sh
## Author <julien@vaubourg.com>

if [ $UID -ne 0 ]; then
        echo -e "ERREUR:\n\tVous devez avoir les droits root" >&2
        exit 1
fi

if [ $# -ne 2 -o "$1" == "-h" ]; then
        echo -e "\n\tRenomme le VServer <old> en <new> ainsi que sa partition."

        echo -e "\n\tNOTE: Ce script stoppe le VServer le temps de l'execution."

        echo -e "\n\tUSAGE"
        echo -e "\t\tvserverlv-rename old new"

        echo -e "\n\tAUTEUR"
        echo -e "\t\tJulien VAUBOURG <julien@vaubourg.com>\n"

        [ $# -ne 2 ] && exit 1 || exit 0
fi

old=$1
new=$2

if [ ! -d /etc/vservers/$old ]; then
        echo -e "ERREUR:\n\tLe VServer $old n'existe pas." >&2
        exit 1
fi

if [ -z "$(vserver $old status | grep stopped)" ]; then
        started=1
        vserver $old stop
else
        started=0
fi

echo -e "\n### Renommage de la partition"

umount /var/lib/vservers/$old
mv /var/lib/vservers/{$old,$new}

lvrename vgRoot var_lib_vservers_{$old,$new}

sed -e "s/var_lib_vservers_$old\([ \t]*\)/var_lib_vservers_$new\1/g"\
 -e "s/\/var\/lib\/vservers\/$old\([ \t]*\)/\/var\/lib\/vservers\/$new\1/g"\
 -i /etc/fstab

echo -e "\n### Renommage du VServer"

mv /etc/vservers/{$old,$new}
echo $new > /etc/vservers/$new/name
ln -sf /etc/vservers/.defaults/vdirbase/$new /etc/vservers/$new/vdir

echo -e "\n### Lancement du VServer nouvellement nomme"

mount /var/lib/vservers/$new
sed "s/^$old\./$new./" -i /var/lib/vservers/$new/etc/hostname

vserver $new start
vserver $new exec /etc/init.d/hostname.sh

[ ! $started ] && vserver $new stop

exit 0
```

FIG. K.1 – [Système principal], fichier ***__HOST/root/bin/vserverlv-rename***

L. Script vserverlv-remove

```
#!/bin/bash
## Author <julien@vaubourg.com>

if [ $UID -ne 0 ]; then
        echo -e "ERREUR:\n\tVous devez avoir les droits root" >&2
        exit 1
fi

if [ -z "$(which vserver)" ]; then
        echo -e "ERREUR:\n\tLes outils VServer ne semblent pas disponibles.\n\tVerifiez que vous etes bien sur le systeme principal et que le PATH est
        exit 1
fi

if [ $# -ne 1 -a $# -ne 2 -o "$1" == "-h" ]; then
        echo -e "\n\tSupprime un VServer ainsi que sa partition."
        echo -e "\tLa taille de celle-ci est donc reportee sur la partition des VServers."
        echo -e "\tToutes les donnees seront perdues !"

        echo -e "\n\tNOTE: Ce script stoppe le VServer le temps de l'execution."

        echo -e "\n\tUSAGE"
        echo -e "\t\tvserverlv-remove -h  (affiche la presente aide)"
        echo -e "\t\tvserverlv-remove vserver"
        echo -e "\t\tvserverlv-remove vserver -f  (supprime les confirmations etape par etape)"

        echo -e "\n\tAUTEUR"
        echo -e "\t\tJulien VAUBOURG <julien@vaubourg.com>\n"

        [ $# -ne 2 ] && exit 1 || exit 0
fi

lvName=$1
[ "$2" == "-f" ] && forceYes=1

if [ ! -d /etc/vservers/$lvName ]; then
        echo -e "ERREUR:\n\tLe VServer $lvName n'existe pas." >&2
        exit 1
fi

## Nom du groupe de lv
vg=vgRoot

## Chemin des VServers, qui servira pour construire le nom de la partition
lvPath=/var/lib/vservers

action() { ((i++))

        if [ -z "$forceYes" ]; then
                echo -e "\n($i/${#cmd[*]}) $1"
                echo -n "Executer : <$2> (Y/n) "
                read execmd
        fi

        if [ -z "$execmd" -o "$execmd" == "y" -o "$execmd" == "Y" ]; then
                [ -z "$forceYes" ] || echo -e "\n### $1..."

                eval "$2"
        else
                echo "NON"
        fi
}

## Supprime le slash en prefixe
lvBaseName=${lvPath#/*}

## Remplace les slashes par des underscores, pour le nom du lv
lvBaseName=${lvBaseName//\//_}

## Tente de determiner la taille actuelle de la partition des VServers
partSize=$(lvs 2> /dev/null | grep -E "[[:space:]]+${lvBaseName}[[:space:]]+")
partSize=$(echo $partSize) # trim
partSize=${partSize##* } # 00.00X
partSize=${partSize%.*} # 00

## Taille de la partition a supprimer
lvSize=$(lvs 2> /dev/null | grep -E "[[:space:]]+${lvBaseName}_$lvName[[:space:]]+")
lvSize=$(echo $lvSize) # trim
lvSize=${lvSize##* } # 00.00X
lvSize=${lvSize%.*} # 00
lvSize=$[partSize + lvSize]

echo -e "Nouvelle taille de $vg-$lvBaseName : ${lvSize}G"

## VServers en fonctionnement
vservers=$(vserver-stat 2> /dev/null | awk 'NR != 1 { printf(" %s", $NF) }')
```

M. Config. `sshd_config`

```
Port 2280
ListenAddress 192.168.0.201
Protocol 2

HostKey /etc/ssh/ssh_host_rsa_key
HostKey /etc/ssh/ssh_host_dsa_key

UsePrivilegeSeparation yes
KeyRegenerationInterval 3600
ServerKeyBits 768
SyslogFacility AUTH
LogLevel INFO

LoginGraceTime 120
PermitRootLogin no
StrictModes yes

RSAAuthentication yes
PubkeyAuthentication yes
AuthorizedKeysFile %h/.ssh/authorized_keys

IgnoreRhosts yes
RhostsRSAAuthentication no
HostbasedAuthentication no
PermitEmptyPasswords no

ChallengeResponseAuthentication no
PasswordAuthentication no

X11Forwarding no
X11DisplayOffset 10
PrintMotd no
PrintLastLog yes
TCPKeepAlive yes

AcceptEnv LANG LC_*
Subsystem sftp /usr/lib/openssh/sftp-server

UsePAM yes
AllowUsers www
```

FIG. M.1 – [VServer www], fichier ***__WWW/etc/ssh/sshd_config***

N. Service `svnserve`

```
#!/bin/sh

set -e
if [ -x /usr/bin/svnserve ] ; then
        HAVE_SVNSERVE=1
else
        echo "Svnserve not installed."
        exit 0
fi

. /lib/lsb/init-functions

case "$1" in
  start)
        log_action_begin_msg "Starting SVN server"
        start-stop-daemon --start --chuid svn:svn --exec /usr/bin/svnserve -- -d -r /var/svn --listen-port 3691
        log_action_end_msg $?
        ;;
  stop)
        log_action_begin_msg "Stoping SVN server"
        start-stop-daemon --stop --exec /usr/bin/svnserve
        log_action_end_msg $?
        ;;
  force-reload|restart)
        $0 stop
        $0 start
        ;;
  *)
        echo "Usage: /etc/init.d/svnserve {start|stop|restart|force-reload}"
        exit 1
        ;;
esac

exit 0
```

FIG. N.1 – [VServer svn], fichier ***__SVN/etc/init.d/svnserve***

O. Config. `sshd_config`

```
Port 3690
ListenAddress 192.168.0.203
Protocol 2

HostKey /etc/ssh/ssh_host_rsa_key
HostKey /etc/ssh/ssh_host_dsa_key

UsePrivilegeSeparation yes
KeyRegenerationInterval 3600
ServerKeyBits 768
SyslogFacility AUTH
LogLevel INFO

LoginGraceTime 120
PermitRootLogin no
StrictModes yes

RSAAuthentication yes
PubkeyAuthentication yes
AuthorizedKeysFile %h/.ssh/authorized_keys

IgnoreRhosts yes
RhostsRSAAuthentication no
HostbasedAuthentication no
PermitEmptyPasswords no

ChallengeResponseAuthentication no
PasswordAuthentication no

X11Forwarding no
X11DisplayOffset 10
PrintMotd no
PrintLastLog yes
TCPKeepAlive yes

AcceptEnv LANG LC_*
Subsystem sftp /usr/lib/openssh/sftp-server

UsePAM yes
```

FIG. O.1 – **[VServer svn]**, fichier ***__SVN/etc/ssh/sshd__config***

P. Script `svn-add`

```
#!/bin/bash
## Author <julien@vaubourg.com>

echo -n "Nom du projet (alnum minus. et -) : "
read project

projectUnix=$(echo $project | tr '[A-Z]' '[a-z]')
projectUnix=$(echo $projectUnix | sed 's/[^[:alnum:]-]*//g')

if [ "$project" != "$projectUnix" ]; then
        project=$projectUnix
        echo "  * Nom corrige : $project"
fi

echo -n "Nom d'utilisateur : "
read user

if ! grep -c ^$project: /etc/group > /dev/null; then
        addgroup --system $project > /dev/null
        echo "Groupe UNIX cree."
else
        echo "Groupe UNIX existant."
fi

if [ ! -d /var/svn/$project ]; then
        svnadmin create /var/svn/$project > /dev/null
        chgrp -R $project /var/svn/$project
        chmod -R 660 /var/svn/$project
        find /var/svn/$project -type d -exec chmod ug+x {} \+
        echo "Projet SVN cree."
else
        echo "Projet SVN existant."
fi

if ! grep -c ^$user: /etc/passwd > /dev/null; then
        useradd -m -k /etc/skel -g $project $user > /dev/null
        echo "Utilisateur UNIX cree (sauf public key !)."
else
        adduser $user $project > /dev/null
        echo "Utilisateur UNIX ajoute au groupe UNIX."
fi

exit 0
```

FIG. P.1 – [VServer swssh], fichier ***__SVN/root/bin/svn-add***

Q. Config. sshd_config

```
Port 443
ListenAddress 192.168.0.202
Protocol 2

HostKey /etc/ssh/ssh_host_rsa_key
HostKey /etc/ssh/ssh_host_dsa_key

UsePrivilegeSeparation yes
KeyRegenerationInterval 3600
ServerKeyBits 768
SyslogFacility AUTH
LogLevel INFO

LoginGraceTime 120
PermitRootLogin no
StrictModes yes

RSAAuthentication yes
PubkeyAuthentication yes
AuthorizedKeysFile %h/.ssh/authorized_keys

IgnoreRhosts yes
RhostsRSAAuthentication no
HostbasedAuthentication no
PermitEmptyPasswords no

ChallengeResponseAuthentication no
PasswordAuthentication no

X11Forwarding no
X11DisplayOffset 10
PrintMotd no
PrintLastLog yes
TCPKeepAlive yes

AcceptEnv LANG LC_*
Subsystem sftp /usr/lib/openssh/sftp-server

UsePAM yes
AllowTcpForwarding yes
```

FIG. Q.1 – [VServer swssh], fichier ***__SWSSH/etc/ssh/sshd_config***

R. Script add-user

```
#!/bin/sh
## Author <julien@vaubourg.com>

echo -n "Nom : "
read nom

useradd -m -k /etc/skel $nom

chmod 700 /home/$nom
chown root:root /home/$nom/.ssh/authorized_keys
chmod 644 /home/$nom/.ssh/authorized_keys

exit 0
```

FIG. R.1 – [VServer swssh], fichier ***__SWSSH/root/bin/add-user***