Cloud Computing

Cluster WebDAV Haute Disponiblité

Jérémy BARBÉ

Décembre 2011

Sommaire

1. **Pré-requis :**

- Un *hyperviseur* (de préférence VMWare / Vsphere) **fonctionnel** avec la configuration minimum suivante :
    - Espace disque : 100GB
    - Mémoire vive : 2GB
    - Processeur : IntelCore 2 Duo cadencé à 2Ghz
    - Réseau : une carte réseau : Fast Ethernet
- Une image ISO (ou CD / DVD) numéro « 1 » des médias d'installation de **Debian 6.0.3 « *Squeeze* » en version 32 bits**

1/ Disposer de 3 adresses IP :

- Une adresse IP pour le serveur (hôte / nœud) « maître » WebDAV (qui sera « l'opérationnel » en conditions « normales »)
- Une adresse IP pour le serveur (hôte / nœud) « esclave » WebDAV (dédié à prendre le relais du nœud « maître » en cas d'incidents sur ce dernier)
- Une adresse IP qui sera utilisée en tant qu'adresse virtuelle pour désigner notre cluster « WebDAV HA »

2/ Disposer de 3 entrées DNS correspondants à chaque IP définie ci-dessus (pour éviter de travailler constamment en adressage IP)

- Créer ces entrées DNS sur votre contrôleur de domaine ou, si non présent, dans le fichier « hosts » de votre poste qui sera utilisé pour la configuration et les tests

Les deux points ci-dessus donnent, dans ce TP, la configuration suivante (sur le réseau 172.20.160.0/20) :

- Webdav01.corp.lan : 172.020.160.013
- Webdav02.corp.lan : 172.020.160.014
- Webdav.corp.lan : 172.020.160.015

Serveur WebDAV HA Master (test JBARBE, Debian 6.0.3) - webdav01.corp.lan	172.020.160.013
Serveur WebDAV HA Slaver (test JBARBE, Debian 6.0.3) - webdav02.corp.lan	172.020.160.014
Adresse IP cluster WebDAV HA (test JBARBE) - webdav.corp.lan	172.020.160.015

2. Installation des serveurs sous Debian 6.0.3

- Commencer par créer **deux** machines virtuelles avec la configuration suivante (conf. minimaliste pour les besoins du TP) : mono *vProcesseur*, 512Mb RAM, 2 cartes réseau « Fast Ethernet », un disque primaire de 15GB

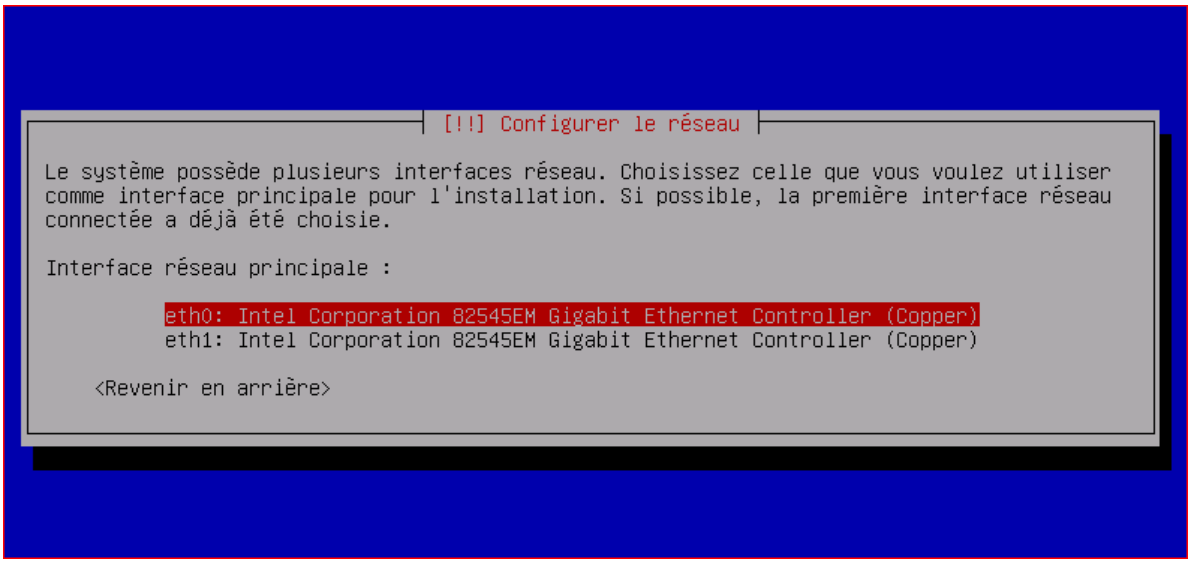
- **Choisir de ne pas connecter vos cartes réseau pour le moment (équivalant « physique » : vos cartes réseau sont installées dans le serveur, mais le câble n'est pas connecté)**

- Votre image ISO ou CD/DVD devra être utilisable par vos machines virtuelles pour procéder à l'installation de Debian

Installation de Debian *Squeeze :*

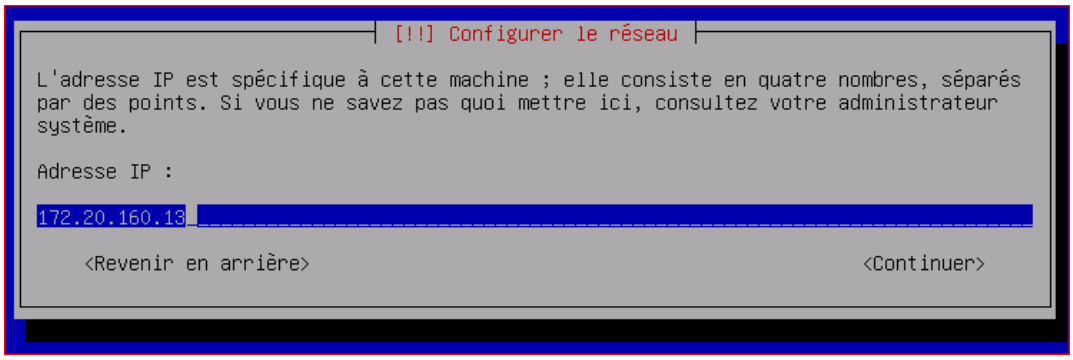
Suivre les étapes ci-dessous pour l'installation de Debian. Il vous faudra répéter la procédure pour l'installation de votre second serveur WebDAV…

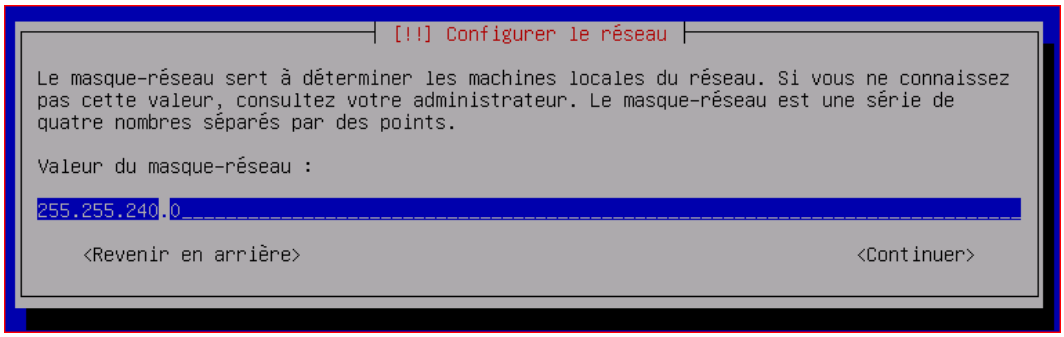
Choisir « Install » et suivre les écrans d'installation….

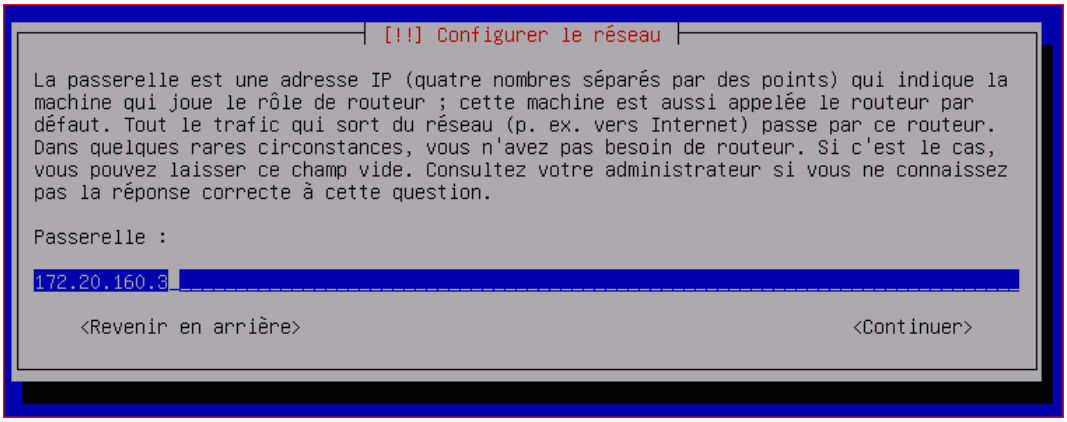


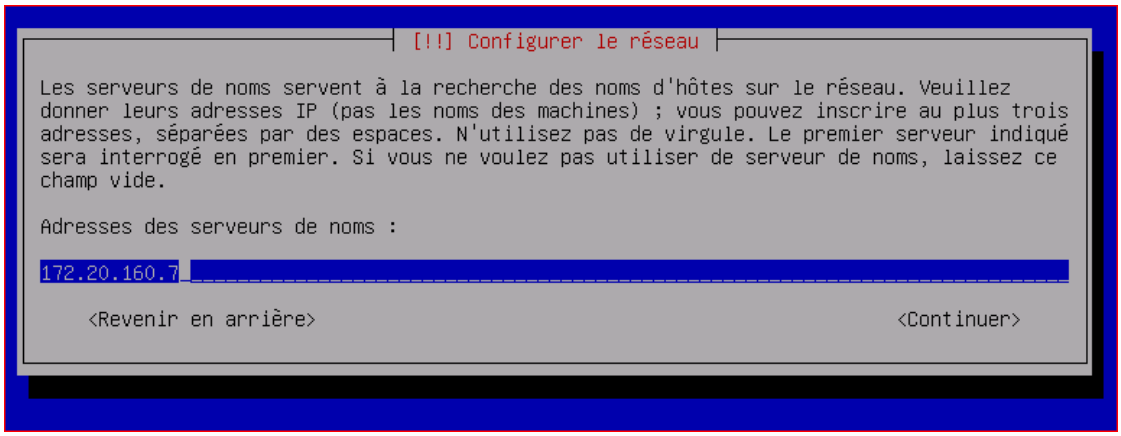
A l'étape de la détection du matériel, vérifier que vos deux cartes réseau sont bien détéctées

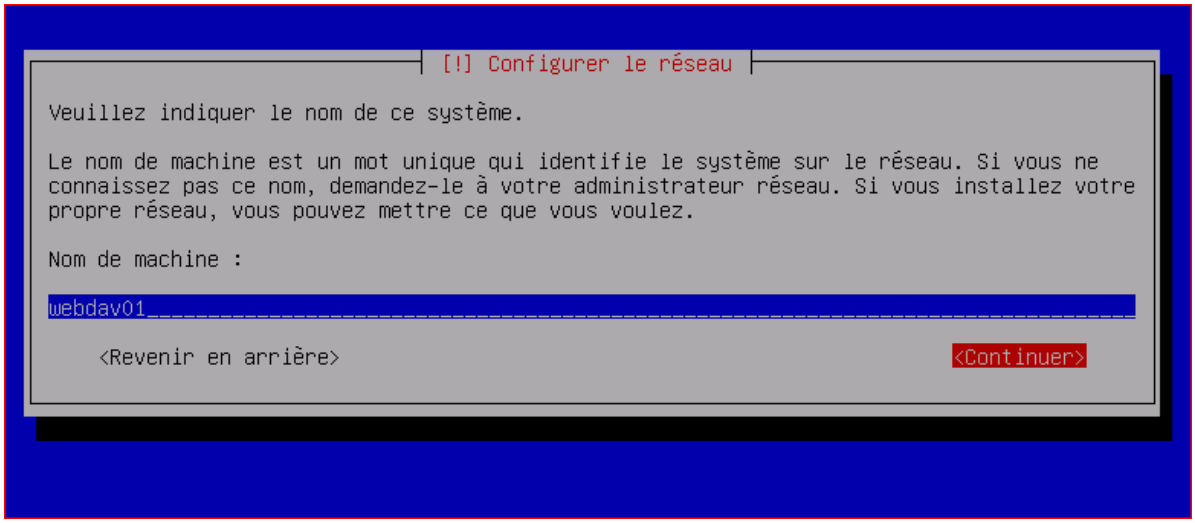
Choisir de configurer le réseau vous-même et définissez vos paramètres IP :









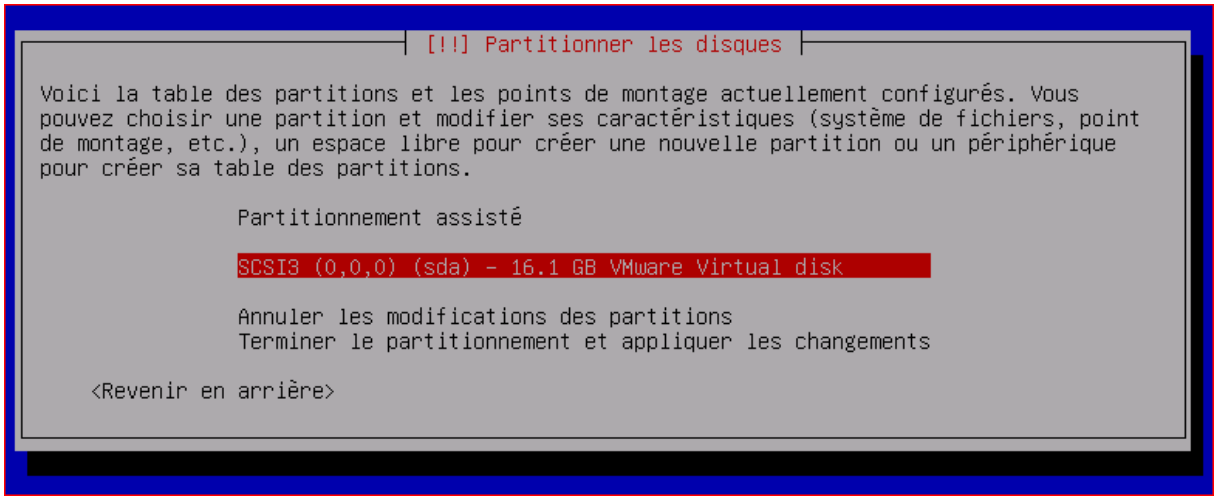


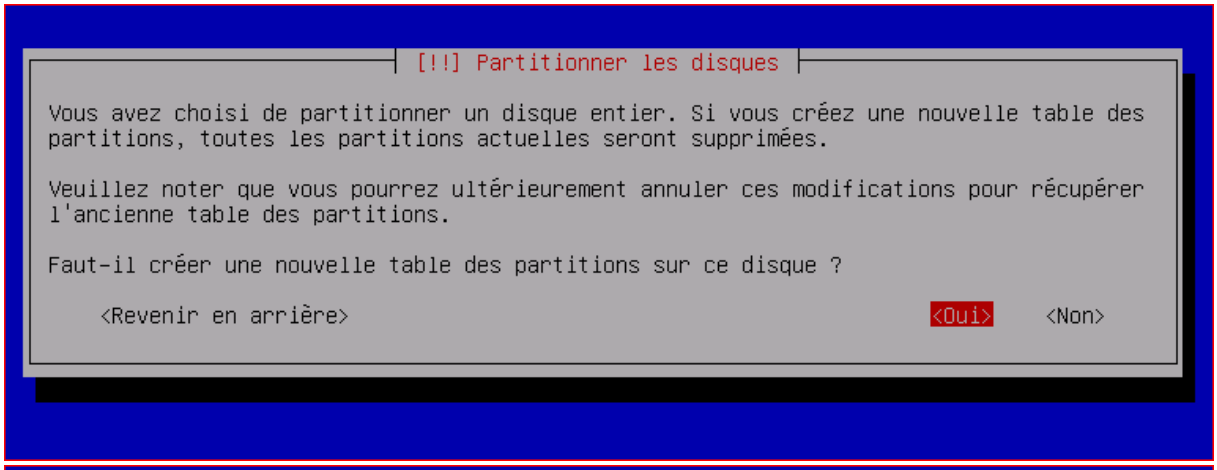
*Attribuer un nom à votre machine (**pensez à changer pour la deuxième !!**)*

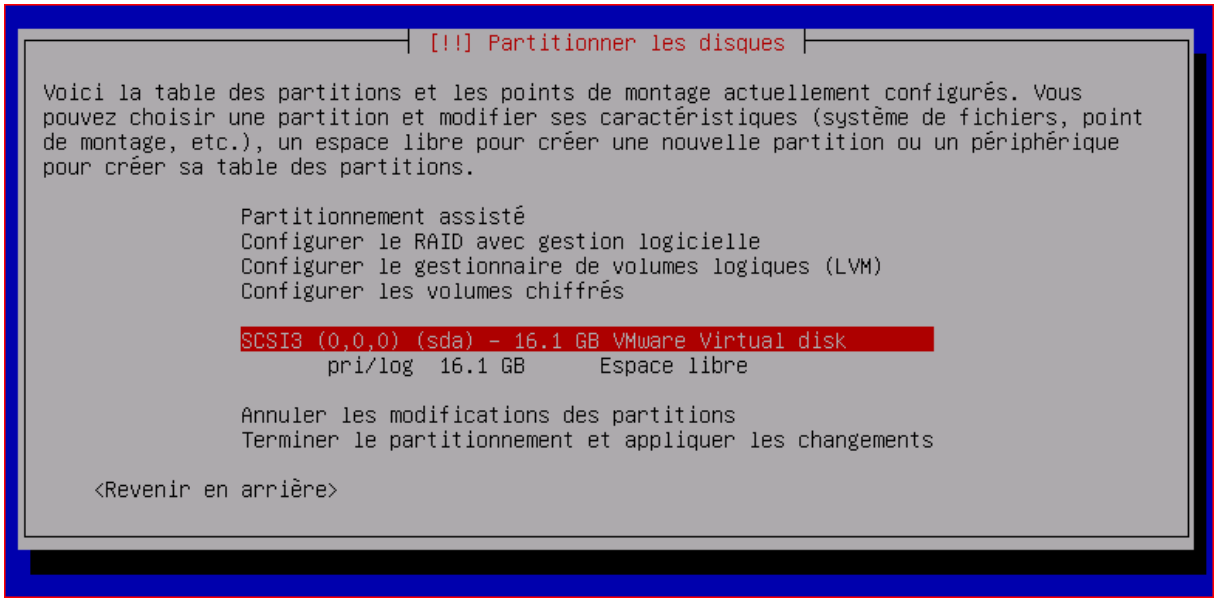
- Définir ensuite le compte « root »
- Créer ensuite un nouvel utilisateur (chez moi, *debianadmin*)

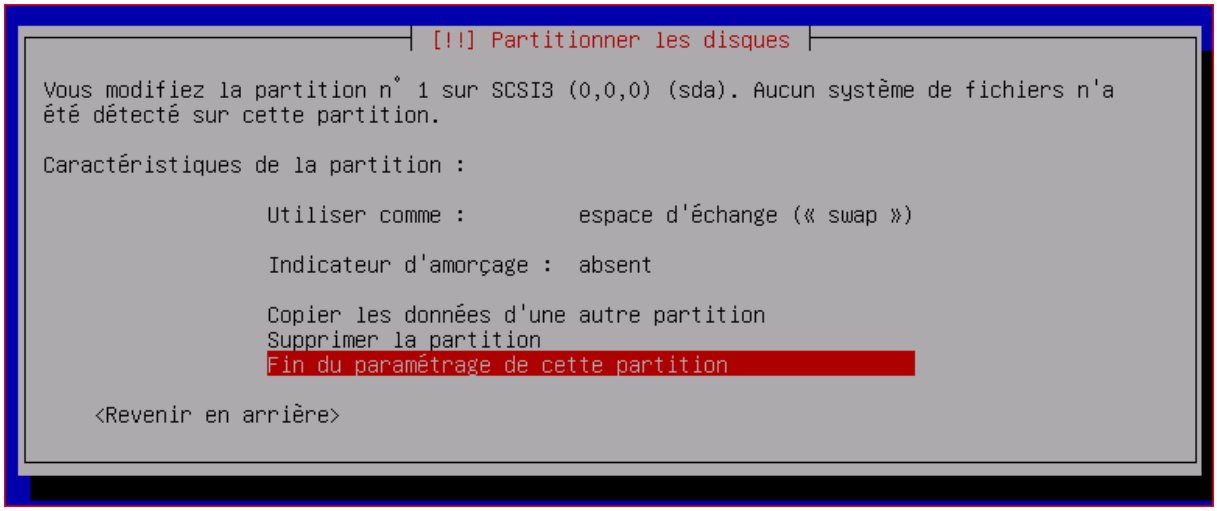
Plan de partitions :

A réaliser « manuellement » :

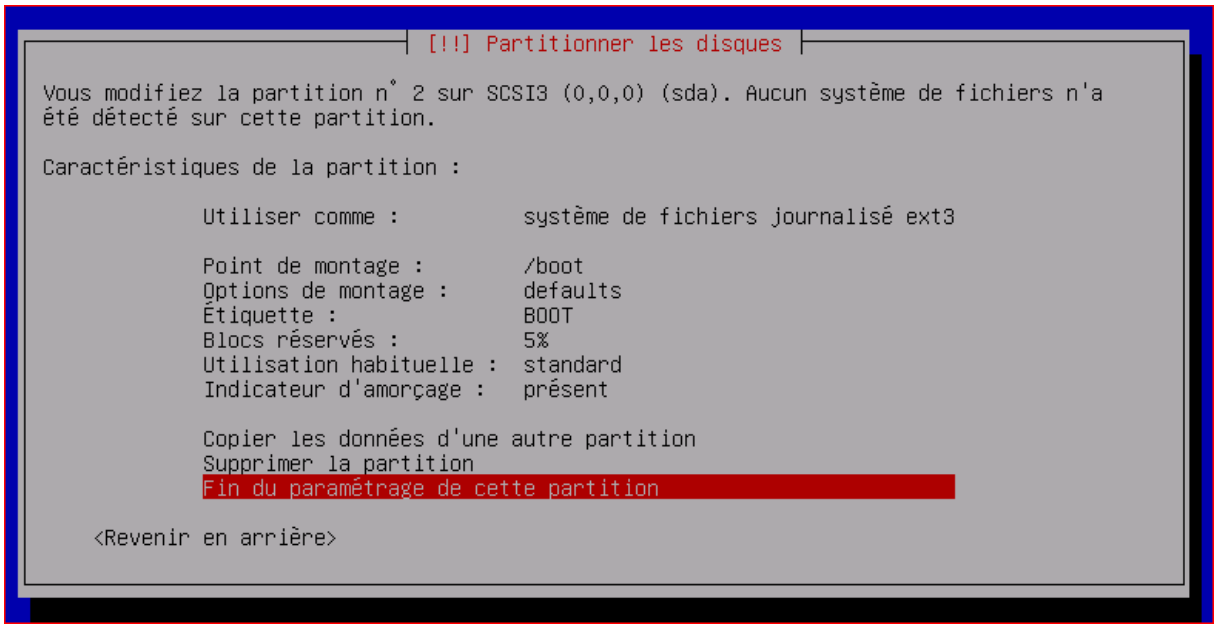




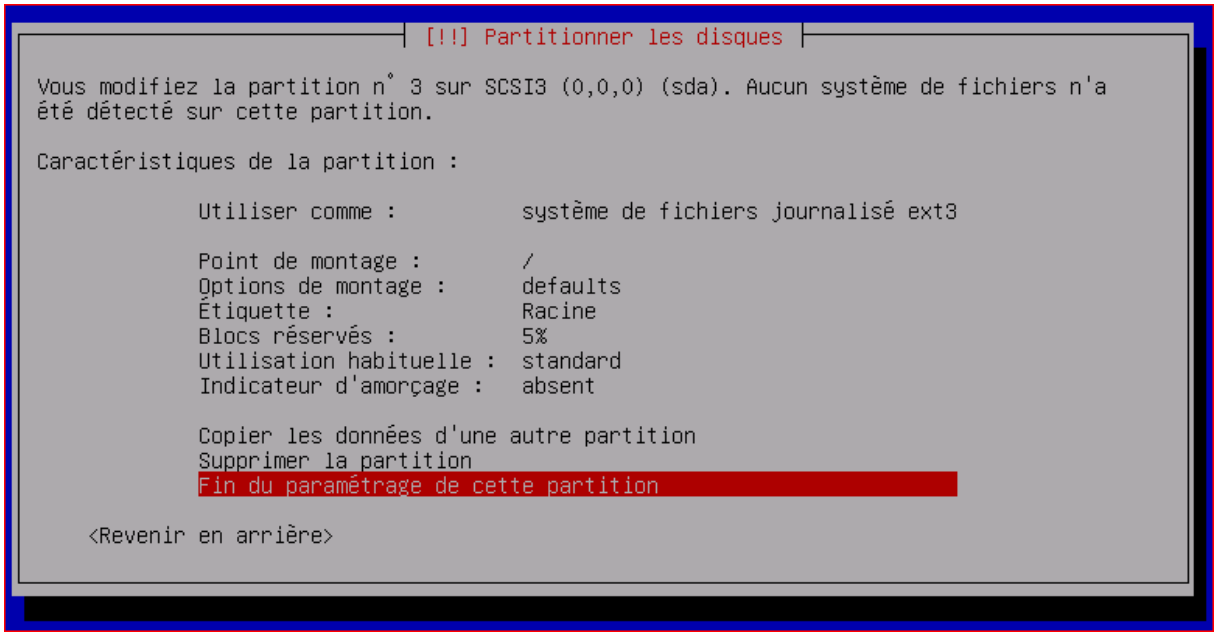




Première partition (début de disque ») : SWAP (= 2x RAM)

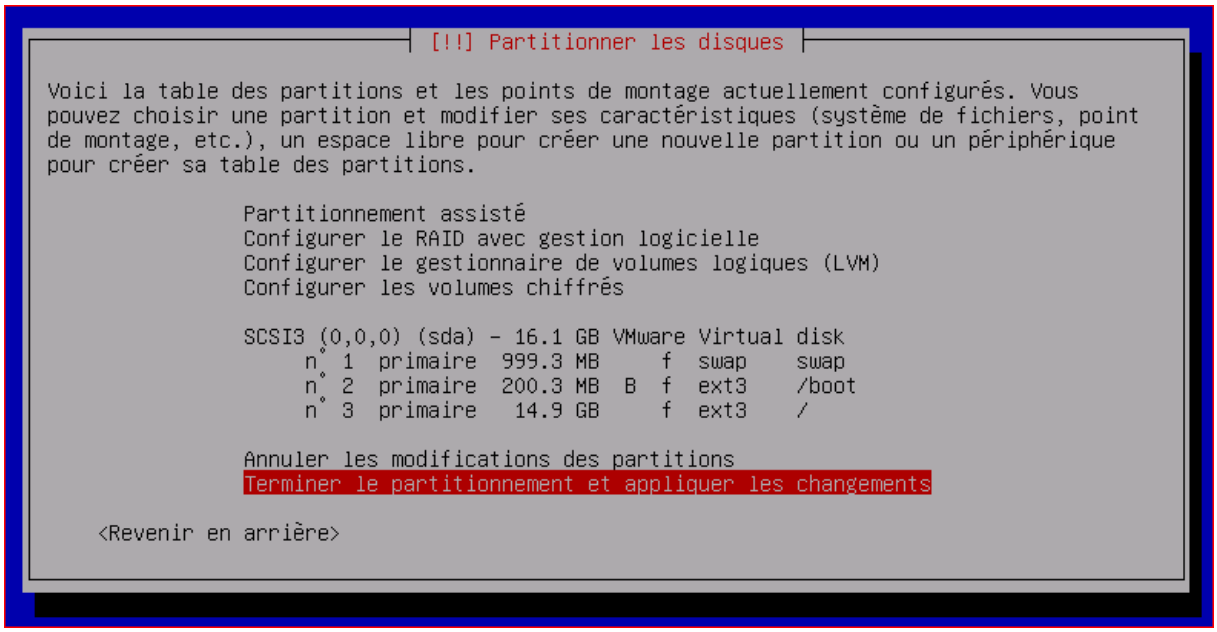


Seconde partition : celle du boot qui doit être amorçable

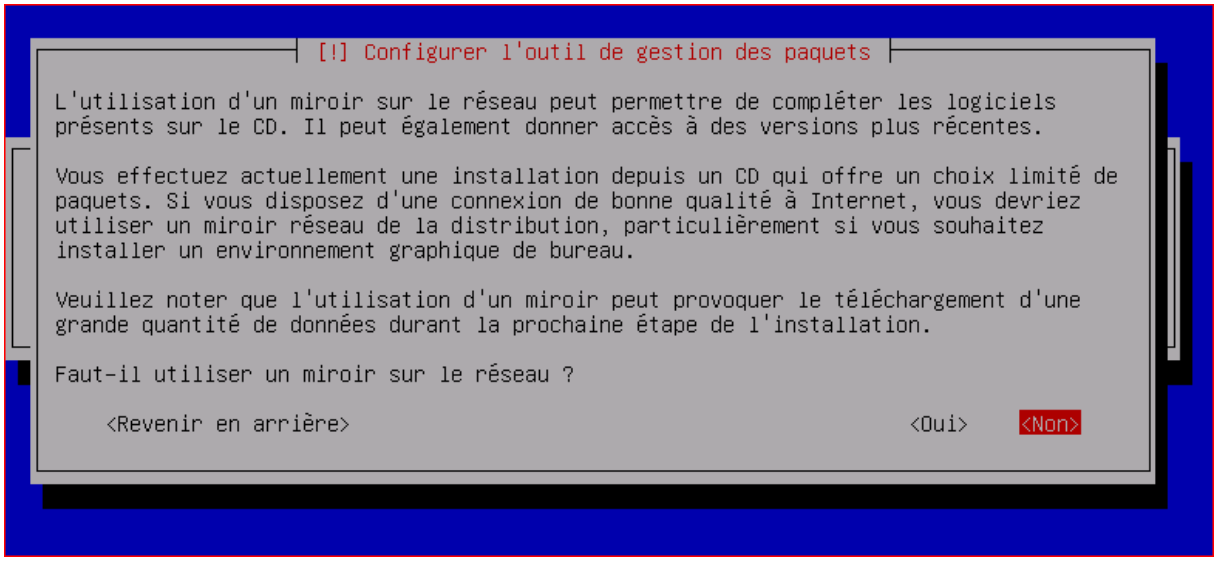


*Troisième et dernière partition : la racine « **/** » (qui contiendra donc le /home, /dev, /var … pour le moment !!)*

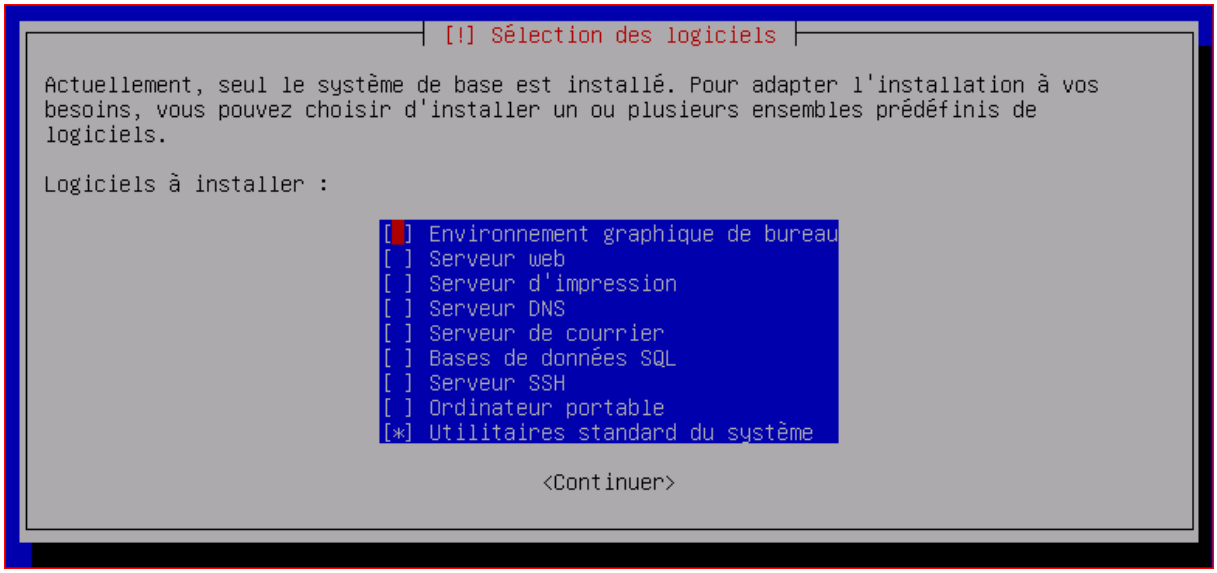
Ce qui donne :



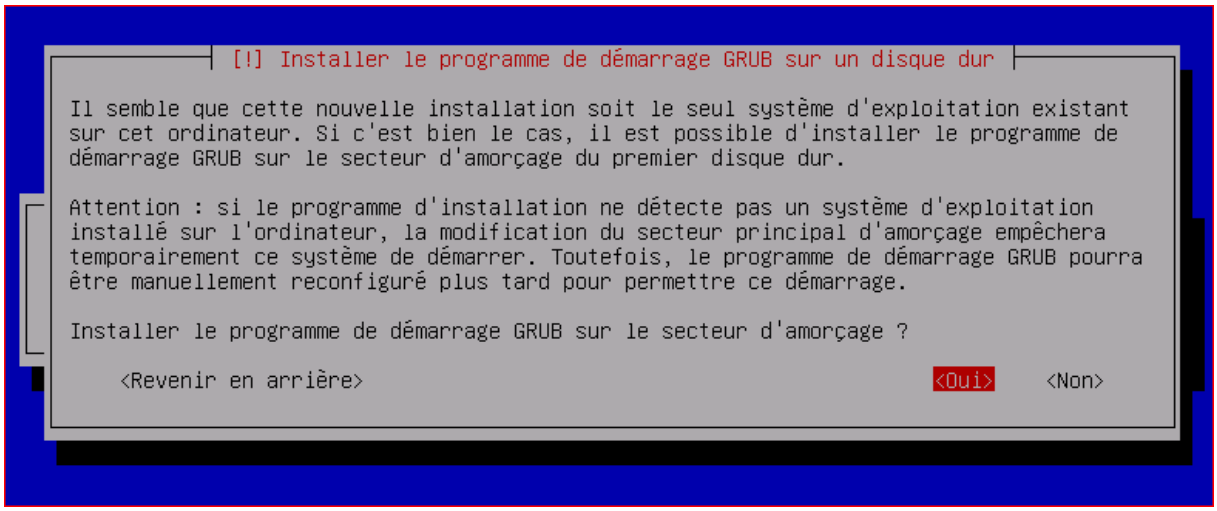
Terminer et appliquer



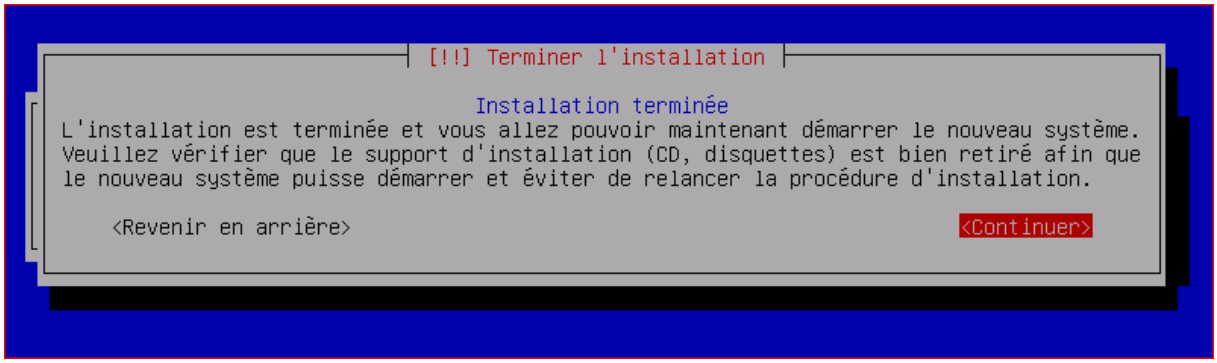
Ne pas configurer de miroir réseau



Ne rien installer d'autre que les utilitaires standard du système ***et surtout pas d'interface graphique !***



Installer GRUB



Déconnecter votre lecteur DVD/CD ou votre image ISO (afin d'éviter de redémarrer sur l'installateur » puis ***continuer***

- **Répéter cette même procédure pour l'installation du deuxième serveur (seul le nom de l'hôte change…)**

3. Configuration « basique » de l'OS : exemple pour le serveur « master »

But : *configuration du réseau, installation et configuration d'un serveur SSH (pour la prise en main distante via un client SSH – Putty -), configuration des outils de virtualisation (optionnel), ajout d'alias pour simplifier l'utilisation des lignes de commandes, colorisation du shell, activation du pavé numérique etc.*

ATTENTION : pour le moment, le seul éditeur texte est VI (VIM sera installé plus tard)…
Voici quelques commandes utiles :

Les commandes de base du mode normal

: q ! [Entrée] pour quitter sans sauver,

: w [Entrée] pour enregistrer,

: wq [Entrée] pour enregistrer et quitter,

: u[Entrée] permet d'annuler (ou :undo).

Les commandes de base du mode insertion

a Ajouter du texte après le curseur

A Ajouter du texte en fin de ligne courante

i Insérer du texte avant le curseur

I Insérer du texte en début de ligne

o ou O Créer une ligne vierge sous ou au dessus du curseur

- Vérifier que le fichier `/etc/hostname` comporte bien une seule ligne avec un nom d'hôte correct

- Vérifier la configuration réseau qui devrait ressembler à ceci : (fichier `/etc/network/interfaces`)

```
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
auto eth0
allow-hotplug eth0
iface eth0 inet static
address 172.20.160.14
netmask 255.255.240.0
gateway 172.20.160.3
dns-nameservers 172.20.160.7 => optionnel
```

⇨ **Remplacer les variables `adress`, `netmask`, `gateway`, `dns-nameservers` par leurs valeurs correctes si besoin.**

Si vous ne disposez pas de serveur DNS, pensez à renseigner les adresse IP et noms DNS de vos deux hôtes dans le fichier `/etc/hosts` et testez avec `uname –n`

- Vérifier le fichier `/etc/resolv.conf` qui devrait ressembler à cela … le modifier ou ajouter des adresses de serveurs DNS au besoin…

```
domain corp.lan
Search corp.lan
nameserver 172.20.160.7
nameserver 172.20.160.8
```

- Créer le `fichier /etc/apt/apt.conf` pour autoriser les mises à jour de paquets en utilisant un serveur proxy pour sortir sur internet (en HTTP)

```
Acquire::http::Proxy "http://adresse-IPde-votre-proxy:PORT";
```

- Éditer les sources des paquets : **`vim /etc/apt/sources.list`** comme il suit :

```
#

# deb cdrom:[Debian GNU/Linux 6.0.3 _Squeeze_ - Official i386 CD Binary-1 20111
08-13:01]/ squeeze main

#deb cdrom:[Debian GNU/Linux 6.0.3 _Squeeze_ - Official i386 CD Binary-1 201110
8-13:01]/ squeeze main

deb http://security.debian.org/ squeeze/updates main
deb-src http://security.debian.org/ squeeze/updates main

# squeeze-updates, previously known as 'volatile'
# A network mirror was not selected during install.  The following entries
# are provided as examples, but you should amend them as appropriate
# for your mirror of choice.
#
deb http://ftp.debian.org/debian/ squeeze-updates main
deb-src http://ftp.debian.org/debian/ squeeze-updates main

deb http://ftp.fr.debian.org/debian/ squeeze main
deb-src http://ftp.fr.debian.org/debian squeeze main_
```

- Suppression du CD / DVD comme source
- Autorisation de l'utilisation des dépôts Internet officiels
- `Editer /etc/wgetrc` pour autoriser le téléchargement de fichiers via un serveur proxy: (si besoin)
  `http_proxy = http://@IP-votre-proxy:PORT/`
- Activer le verrouillage numérique au démarrage :

Editer le fichier `/etc/init.d/bootmisc.sh` et ajouter en fin de fichier :

```
for tty in dev/tty[1-6]
do
 setleds -D +num < $tty > /dev/null
done
```

```
#ajout jbarbe, numlock au boot en mode console
for tty in dev/tty[1-6]
do
 setleds -D +num < $tty > /dev/null
done
```

⇨ **Redémarrez votre serveur afin de bien prendre en compte tous les changements réseau et connectez votre carte réseau virtuelle au réseau (+ tests de base d'accès au réseau…)**

A. Installation de VIM & configuration de VIM:

```
apt-get update
apt-get install vim
```

- Activer la coloration syntaxique de VIM : éditer le fichier `/etc/vim/vimrc` et décommenter : `syntax on`

```
" Vim5 and later versions support syntax highlighting. Uncommenting the next
" line enables syntax highlighting by default.
syntax on
```

- Activer les alias et la couleur du shell :

Dans le `fichier /home/debianadmin/.bashrc` décommenter **ForceColorPrompt=Yes** et les alias.

Copier ensuite le fichier **.bashrc** dans le répertoire **root** via `cp /home/debiandamin/.bashrc /root/.bashrc`. *Se relogguer.*

B. Installation de SSH : `apt-get install ssh`

C. Mettre à jour la distribution & installer les vmware tools *:*

```
apt-get update
apt-get install make gcc
apt-get install linux-headers-$(uname -r)
mount /dev/cdrom/ /mnt
cp /mnt/VmWareTools.tar.gz /tmp/
tar –zxvf /tmp/VmWareTools.tar.gz
cd /tmp/WmareTools/
./vmware-install.pl
apt-get upgrade
```

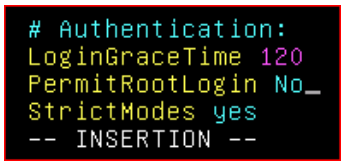
⇨ **Répéter cette même procédure pour la configuration basique du deuxième serveur (en modifiant les paramètres variables…)**

⇨ **Tester le dialogue entre les deux serveurs WebDAV (« master » et « slave »)**

4. Sécurisation de vos serveurs

A. *Sécurisation de **SSHd***

- Ne pas permettre la connexion directe en SSH en tant que *root* est indispensable
- N'autorisez que le strict minimum d'adresses IP à se connecter en SSH sur vos hôtes



Interdire l'accès **root** par SSH : modifier `/etc/ssh/sshd_config` : passer `PermitRootLogin` à **NO**

N'autoriser que certaines IP à se connecter en SSH : utiliser les fichiers `/etc/hosts.allow` et `/etc/hosts.deny`.

Dans `/etc/hosts.deny` : Interdire toute connexion SSH

- `sshd:ALL`

Dans `/etc/hosts.allow` : autoriser uniquement les IP voulues (poste d'administration + serveurs WebDAV)

- `sshd:192.168.1.5`
- `sshd:202.54.1.5`

Pour plus de souplesse, et, cela sera utile plus tard dans l'activité, il serait bon d'autoriser vos serveurs WebDAV à faire des connexions SSH entre eux (pour faire du *Rsync* ou des transferts *SCP* en *turbo SSH* par exemple…)
Cela est possible grâce à l'argument « *Match* » qui permet de faire des conditionnelles dans le fichier `/etc/ssh/sshd_config`

Ici, j'autorise mes deux serveurs WebDAV et deux postes « administrateurs » à faire une connexion SSH directe en tant que *root*

```
# Ajouts JBARBE
# Permettre la connexion SSH en tant que root pour certaines adresses IP
Match Address 172.20.160.13,172.20.160.14,172.20.160.54,172.20.160.58
        PermitRootLogin yes
```

Attention : l'argument « Match » englobe les arguments suivants (sauf si appel à une nouvelle conditionnelle de type « Match ») : placer donc systématiquement cet argument en fin de fichier….

B. *Protection contre les attaques DoS et le « Brute-Force » :*

Afin d'augmenter d'un cran la sécurité de vos serveurs, il convient d'installer une solution qui peut bannir tout hôte pouvant présenter une menace pour le serveur, comme une attaque *DoS* , un brute- force sur les mots de passe, des échecs d'authentification etc.

Fail2Ban fait ça très bien et est simple à administrer.

Installation : `apt-get install fail2ban`

A savoir : Les fichiers de configuration se trouvent dans `/etc/fail2ban`
- `fail2ban.conf` : le fichier de configuration général
- `jail.conf` : le paramétrage des différents services à protéger
- `filter.d` : répertoire contenant les motifs à détecter dans les fichiers de logs (par services)
- `action.d` : répertoire contenant les différentes actions à entreprendre

⇨ Après chaque modification, il conviendra de redémarrer le service Fail2Ban pour appliquer les changements dans les différents fichiers de configuration…

ATTENTION :

- ✓ pensez à définir une adresse IP qui sera ignorée par **Fail2Ban** pour pouvoir continuer à prendre la main sur votre serveur en cas d'erreur de configuration
- ✓ je vous conseille également d'ignorer l'adresse IP de vos deux serveurs **WebDAV** ainsi que l'IP virtuelle utilisée par **Heartbeat**

```
[DEFAULT]

# "ignoreip" can be an IP address, a CIDR mask or a DNS host
ignoreip = 127.0.0.1

# Adresse IP d'un poste d'administration
ignoreip = 172.20.160.54
#adresses IP des hotes + IP virtuelle du cluster HA
ignoreip = 172.20.160.13
ignoreip = 172.20.160.14
ignoreip = 172.20.160.15

# Duree de l'interdiction de dialogue en secondes
bantime  = 600 #10 min

# Nombres maximum d'echecs de connexion sur le serveur
maxretry = 3
```

- **Commencez par sécuriser votre service *sshd*. Une fois Apache + WebDAV installés, vous devrez également sécuriser ces services.**

⇨ **A faire sur vos deux serveurs…**

Voir aussi : **http://j2c.org/informatique/linux/fail2ban.php**

Commandes utiles :
- Voir les IP bloquées en fonction des filtres : `iptables –L`
- Voir toutes les IP bloquées : `iptables –L | grep « DROP »`
- Déloquer toutes les IP bannies (temporairement dans « IPTABLES ») : le plus simple et de redémarrer Fail2Ban : `invoke-rc.d fail2ban restart`
- Débloquer une IP : `iptables –D nom-du-filtre-ip-bannie numéro-IP`

Exemples :

```
root@dav-node1:~# iptables -L | grep "DROP"
DROP       all  --  irh-tmg.corp.lan     anywhere
DROP       all  --  vostro19.corp.lan    anywhere
```

Ces hôtes sont bloqués

```
Chain fail2ban-apache-noscript (1 references)
target     prot opt source               destination
DROP       all  --  irh-tmg.corp.lan     anywhere
DROP       all  --  vostro19.corp.lan    anywhere
RETURN     all  --  anywhere             anywhere
```

Ces hôtes sont bloqués pour tentative d'exécution d'un script non présent ou non autorisé

```
root@dav-node1:~# iptables -D fail2ban-apache-noscript 2
root@dav-node1:~# iptables -L | grep "DROP"
DROP       all  --  irh-tmg.corp.lan     anywhere
root@dav-node1:~#
```

La deuxième IP bannie a été retirée du filtre IPTABLES « fail2ban-apache-noscript » : elle est de nouveau autorisée

Fail2Ban et alertes par Email : Fail2Ban peut vous informer de chaque action qu'il effectue par mail. Pour cela, il faudra soit installer un serveur SMTP, soit installer Postfix en mode RELAY (Postfix transmettra simplement les messages à envoyer à un serveur SMTP). J'ai, pour ma part, choisi cette option. Cependant, l'installation de postfix entraîne la désinstallation d'EXIM4… qui est lui utilisé par les scripts d'HeartBeat pour alerter par mail en cas de bascule de nœud…

Installation de postfix :

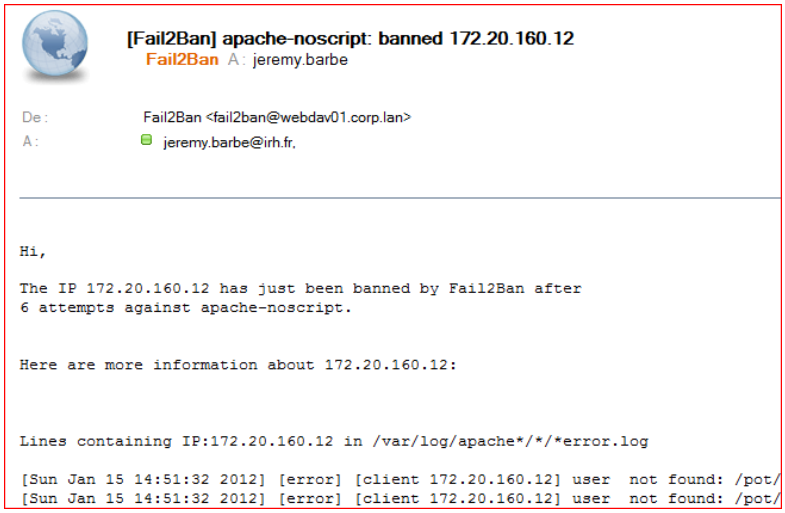
```
Apt-get install postfix
```

- Choisir de l'installer en mode « *Serveur Internet* » ou « *Serveur FAI* » pour préconfigurer le mode « *relais* »
- Dans le fichier `/etc/postfix/main.cf` ajouter/modifier la ligne `relayhost` en y ajoutant l'adresse IP de votre serveur SMTP
- Relancer postfix

```
relayhost = 10.16.22.1

"/etc/postfix/main.cf" 41L, 1319C Ã©crit(s)
```

Voici un aperçu de ce que cela peut donner :



[Fail2Ban] apache-noscript: banned 172.20.160.12
Fail2Ban A : jeremy.barbe

De : Fail2Ban <fail2ban@webdav01.corp.lan>
A : jeremy.barbe@irh.fr,

Hi,

The IP 172.20.160.12 has just been banned by Fail2Ban after
6 attempts against apache-noscript.

Here are more information about 172.20.160.12:

Lines containing IP:172.20.160.12 in /var/log/apache*/*/*error.log

[Sun Jan 15 14:51:32 2012] [error] [client 172.20.160.12] user not found: /pot/
[Sun Jan 15 14:51:32 2012] [error] [client 172.20.160.12] user not found: /pot/

C. *Optimiser le dialogue entre vos serveurs*

Pour le moment, vous pouvez, via SSH, faire communiquer vos deux serveurs : prise en main de la console, transfert de fichiers via SCP.

```
root@webdav01:~# ssh debianadmin@webdav02
The authenticity of host 'webdav02 (172.20.160.14)' can't be established.
RSA key fingerprint is 70:8a:91:f4:c6:2a:50:66:c6:23:88:ef:17:2d:85:9e.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'webdav02' (RSA) to the list of known hosts.
debianadmin@webdav02's password:
Linux webdav02 2.6.32-5-686 #1 SMP Thu Nov 3 04:23:54 UTC 2011 i686

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Thu Jan 12 20:22:40 2012 from vostro19.corp.lan
debianadmin@webdav02:~$
```

Ici, je prends la main sur WebDAV02 à partir de WebDAV01 en SSH

Problème : il faut à chaque fois saisir le mot de passe… et si l'on souhaite automatiser certaines actions entre les deux serveurs, il faudra faire apparaître le mot de passe dans votre script par exemple…

La solution : Turbo SSH permet de faire une connexion entre un ou plusieurs hôtes sans avoir à saisir le mot de passe. Ce système se base sur un système d'échange de clefs publique et privée…

Nous allons permettre une connexion SSH dans les deux sens :

- webDAV01 => WebDAV02
- WebDAV02 => WebDAV01

Il faudra donc, sur chaque serveur, générer une clef privée et copier la clef publique correspondant sur l'hôte qui devra pouvoir se connecter sans saisie de mot de passe.

2 machines :

- **A,** hôte qui doit pouvoir se connecter en mode turbo sur B
- **B,** hôte qui accepte la connexion en mode turbo

ATTENTION : Les clefs sont générées depuis le poste A. Le login en cours est inscrit dans la clef (ex: debianadmin)
Ce login doit donc exister sur le poste B !

Commande sur le poste A : génération des clefs publique et privée :

```
ssh-keygen -t rsa
```

- clef privée : %user%/.ssh/id_rsa
- clef publique : %user%/.ssh/id_rsa.pub

ATTENTION : Ne pas saisir de passphrase lors de la génération des clefs, cela impliquerait de devoir la saisir à la connexion ou au reboot du serveur SSH.

Commande sur le poste B :
La clef publique doit être placée dans le fichier `authorized_keys` (à créer)

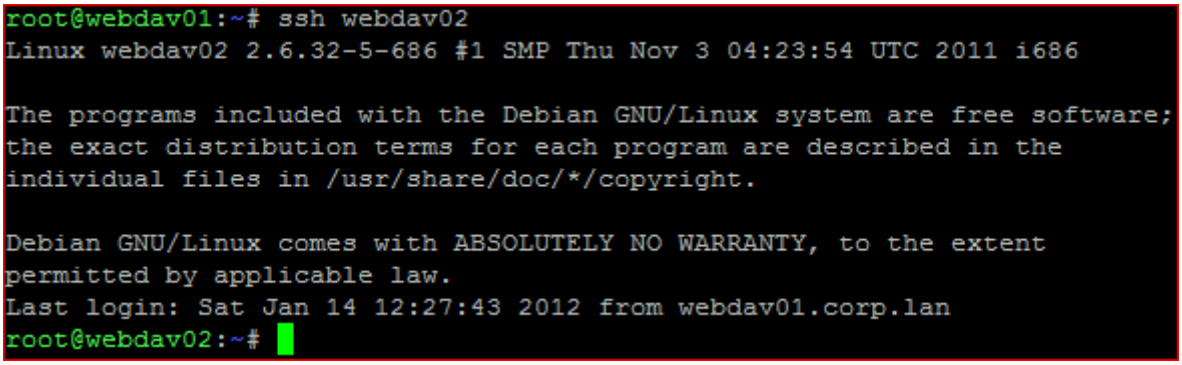
Copie de la clef publique du poste A :

- `scp login@IPPOSTEA:/home/%user%/.ssh/id_rsa.pub /home/%user%/.ssh/authorized_keys`
- `chmod 600 /htome/%user%/.ssh/authorized_keys`

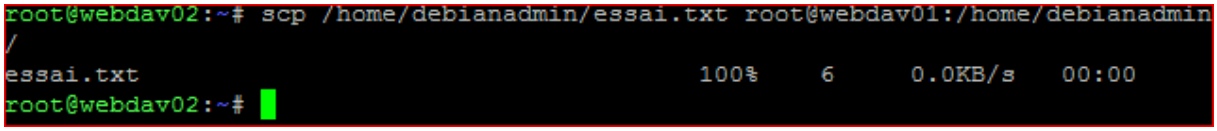
Attention : si une connexion en tant que *root* est nécessaire, il faudra autoriser la connexion SSH en *root* en passant **PermitRootLogin** à **1** dans le fichier `/etc/ssh/sshd_config`

⇨ Dans mon cas, j'ai réglé le problème avec les exceptions via l'argument « match »

⇨ **testez une connexion SSH de WebDAV01 vers WebDAV02 et inversement : si vous n'avez plus besoin de saisir de mot de passe, alors cela fonctionne !**



Connexion SSH de WebDAV01 vers WebDAV02 : plus besoin de mot de passe



Copie SSH d'un fichier de WebDAV02 vers WebDAV01 : plus besoin de mot de passe

5. Installation du serveur web et configuration

A. Installation d'Apache, php, MySQL, WebDAV

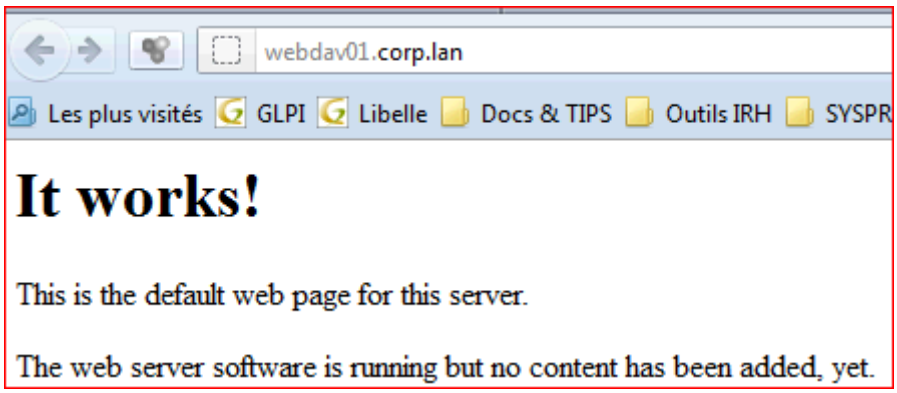
Nous allons installer et configurer, dans un premier temps, un serveur Web basé sur Apache 2. Nous y ajouterons le module WebDAV. En parallèle, nous allons adjoindre à notre serveur le support de php5 et MySQL (car nécessaires dans la phase de mise en place d'une interface graphique sur notre serveur WebDAV)

```
apt-get update
apt-get install apache2 php5 mysql-server php5-mysql phpmyadmin
php5-ldap
```

- Pour MySQL, donner un mot de passe à l'utilisateur *root*
- Pour phpMyAdmin, choisir de l'installer pour *apache2*

Pour le moment, nous avons juste un serveur Apache supportant *php* et une base *MySQL* opérationnelle (et vide)

Testez : avec un navigateur, ouvrir http://webdav01.corp.lan. Cette page devrait s'afficher… (remplacer au besoin avec votre propre nom DNS ou avec l'adresse IP de votre serveur)



Pour tester PHP, il suffit de créer une page **test.php** dans `/var/www/` avec vim. Cette page contiendra le code suivant :

```
<?php
// Affiche toutes les informations relatives à la gestion de php et
ses modules
phpinfo();
?>
```

Avec un navigateur, si vous affichez cette page, vous devriez obtenir ceci :

PHP Version 5.3.3-7+squeeze3

System	Linux webdav01 2.6.32-5-686 #1 SMP Thu Nov 3 04:23:54 UTC 2011 i686
Build Date	Jun 28 2011 13:11:03
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php5/apache2
Loaded Configuration File	/etc/php5/apache2/php.ini
Scan this dir for additional .ini files	/etc/php5/apache2/conf.d
Additional .ini files parsed	/etc/php5/apache2/conf.d/gd.ini, /etc/php5/apache2/conf.d/ldap.ini, /etc/php5/apache2/conf.d/mcrypt.ini, /etc/php5/apache2/conf.d/mysql.ini, /etc/php5/apache2/conf.d/mysqli.ini, /etc/php5/apache2/conf.d/pdo.ini, /etc/php5/apache2/conf.d/pdo_mysql.ini, /etc/php5/apache2/conf.d/suhosin.ini
PHP API	20090626
PHP Extension	20090626
Zend Extension	220090626
Zend Extension Build	API220090626,NTS
PHP Extension Build	API20090626,NTS
Debug Build	no
Thread Safety	disabled
Zend Memory Manager	enabled
Zend Multibyte Support	disabled
IPv6 Support	enabled
Registered PHP Streams	https, ftps, compress.zlib, compress.bzip2, php, file, glob, data, http, ftp, phar, zip
Registered Stream Socket Transports	tcp, udp, unix, udg, ssl, sslv3, sslv2, tls
Registered Stream Filters	zlib.*, bzip2.*, convert.iconv.*, string.rot13, string.toupper, string.tolower, string.strip_tags, convert.*, consumed, dechunk, mcrypt.*, mdecrypt.*

This server is protected with the Suhosin Patch 0.9.9.1
Copyright (c) 2006-2007 Hardened-PHP Project Copyright (c) 2007-2009 SektionEins GmbH

수호신

This program makes use of the Zend Scripting Language Engine:
Zend Engine v2.3.0, Copyright (c) 1998-2010 Zend Technologies
with Suhosin v0.9.32.1, Copyright (c) 2007-2010, by SektionEins GmbH

Powered By Zend Engine 2

Il nous faut maintenant activer WebDAV sur notre serveur. WebDAV est une «extension du protocole http. Sous Apache, il est composé de deux modules : **dav.mod** et **dav_fs.mod**. Ces modules sont installés par défaut en même temps qu'apache mais ne sont pas activés.

A savoir sur apache dans une configuration par défaut :

- Binaire du service : `/usr/sbin/apache2`
- Fichier de configuration principal : `/etc/apache2/apache2.conf`
- Fichier de configuration de la sécurité d'apache : `/etc/apache2/cond.d/security`
- Fichier de configuration du site apache par défaut : `/etc/apache2/site-available/default`
- Répertoire des « sites » apapche : `/etc/apapche2/sites-available/`
- Répertoire des « sites » apache actifs : `/etc/apache2/sites-enabled/`
- Répertoires des modules apache : `/etc/apache2/mods-available/`
- Répertoires des modules apache actifs : `/etc/apache2/mods-enabled/`
- Répertoire de configuration destiné aux modules tiers : `/etc/apache2/conf.d/`
- Emplacement des logs : `/var/log/apache2`
- Répertoire de base des sites Web : `/var/www/`

Apache utilisait initialement le fichier de configuration **httpd.conf**. Dans le but de maintenir une compatibilité avec une configuration préétablie ou avec certains « vieux » modules, ce fichier est maintenu et il est chargé au démarrage d'apache par un appel « `include httpd.conf` » dans le fichier **apache2.conf. Il en est de même pour la majorité des fichiers de** configuration (modules tiers, sites apache etc.) : **phpmyadmin** par exemple dispose d'un fichier de configuration `/etc/apache2/conf.d/phpmyadmin` qui est chargé par l'argument `include conf.d`

L'activation d'un module apache se fait la création d'un lien symbolique dans le répertoire `/etc/apache2/mods-enabled/` pointant vers le module disponible dans le répertoire `/etc/apache2/mods-available/`

Pour simplifier l'activation d'un module apache, plutôt que de créer le lien symbolique avec la commande **`ln -s`** , il suffit d'utiliser la commande `a2enmod` suivie du nom du module à activer.

C'est le même principe pour l'activation d'un site apache, mais on utilise la commande **a2ensite** (on peut également créer le lien symbolique manuellement avec `ln -s`)

Après une modification dans un fichier de configuration, il est nécessaire de demander au service apache de relire l'ensemble des fichiers de configuration pour prendre en compte les changements :

- **`invoke-rc.d apache2 reload`**

Après activation d'un module, il est nécessaire de redémarrer le service apache :

- **`invoke-rc.d apache2 restart`**

Activation des modules WebDAV :

- `a2endmod dav`
- `a2enmod dav_fs`
- `a2enmod dav_lock`

- `invoke-rc.d apache2 restart`

```
root@webdav01:/var/www# a2enmod dav
Enabling module dav.
Run '/etc/init.d/apache2 restart' to activate new configuration!
root@webdav01:/var/www# a2enmod dav_fs
Considering dependency dav for dav_fs:
Module dav already enabled
Enabling module dav_fs.
Run '/etc/init.d/apache2 restart' to activate new configuration!
root@webdav01:/var/www# invoke-rc.d apache2 restart
```

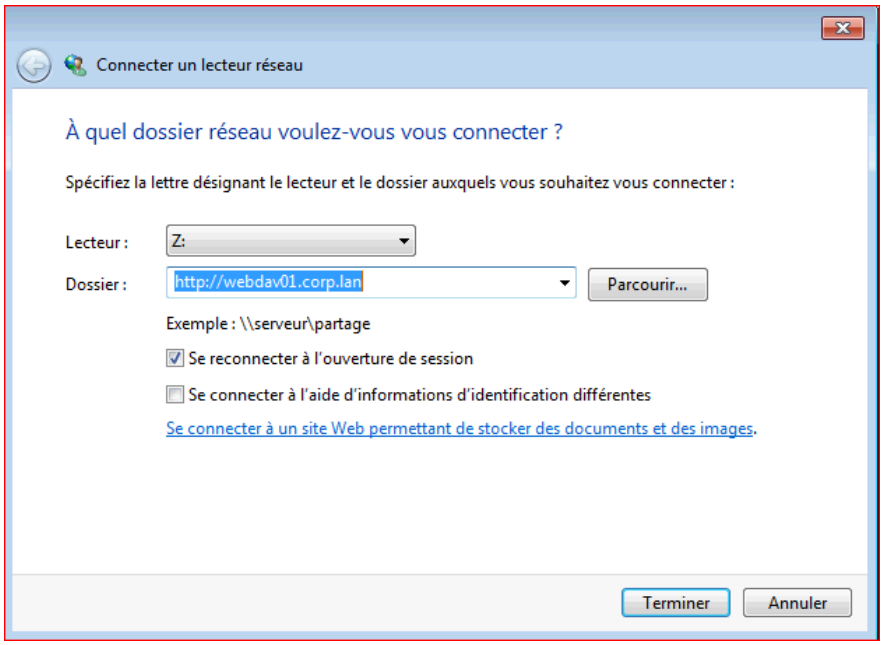
Les modules sont activés

Il ne reste plus qu'à tester : dans un premier temps, on va autoriser l'accès en lecture à notre répertoire Web par défaut avec un client WebDAV.

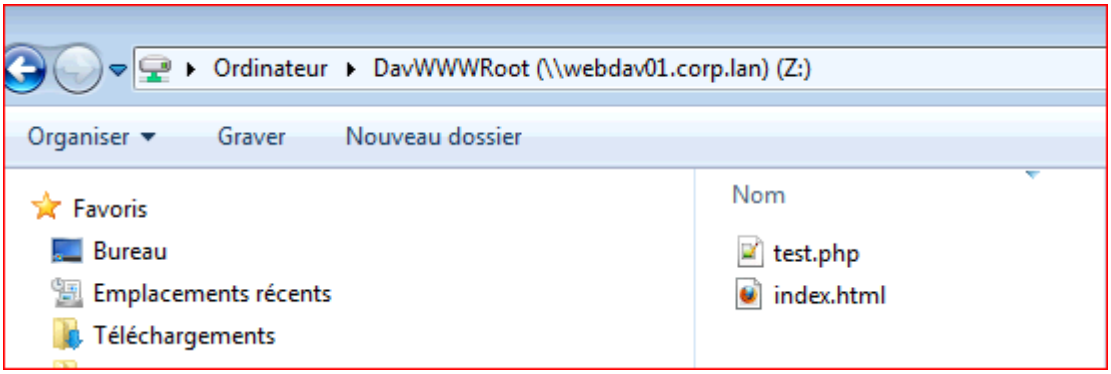
Pour se faire, il suffit d'éditer le fichier de configuration du site par défaut d'apache (`/etc/apache2/sites-available/default`) et d'ajouter l'argument « **DAV on** » dans les réglages de notre directory par défaut (`/var/www/`)

```
<Directory /var/www/>
        Options Indexes FollowSymLinks MultiViews
        AllowOverride None
        Order allow,deny
        allow from all
        # Autorisation d'access WebDAV en lecture seule
        DAV on
</Directory>
```

- Recharger la configuration d'apache et tester !



Le plus simple sur Windows Seven : connecter un lecteur réseau à la ressource http://votreserveurDAV/



L'accès en lecture seule fonctionne

Il va maintenant falloir :

- Sécuriser notre serveur apache
- Configurer notre serveur WebDAV (définir le répertoire de stockage des fichiers, les droits d'accès etc)

B. Configuration et sécurisation du serveur

On va maintenant sécuriser notre serveur apache

Sécurisation du serveur :

Le serveur, lorsqu'il ne peut satisfaire une requête, renvoie à l'internaute une page d'erreur. Cette dernière contient une signature qui, par défaut, indique la version du système d'exploitation, la version du serveur apache, le port d'écoute. Il est nécessaire de minimiser ces informations. On édite donc le fichier **`/etc/apache2/conf.d/security`**

- On remplace **`ServerTokens Full`** par **`ServerTokens Prod`** ce qui permet de donner le minimum d'information dans la signature des pages d'erreur.
- On peut également mettre **`ServerSignature`** à **`Off`**

```
# ServerTokens
# This directive configures what you return as the Server HTTP response
# Header. The default is 'Full' which sends information about the OS-Type
# and compiled in modules.
# Set to one of:  Full | OS | Minimal | Minor | Major | Prod
# where Full conveys the most information, and Prod the least.
#
#ServerTokens Minimal
ServerTokens Prod
#ServerTokens Full

#
# Optionally add a line containing the server version and virtual host
# name to server-generated pages (internal error documents, FTP directory
# listings, mod_status and mod_info output etc., but not CGI generated
# documents or custom error documents).
# Set to "EMail" to also include a mailto: link to the ServerAdmin.
# Set to one of:  On | Off | EMail
#
#ServerSignature Off
ServerSignature Off
```

Protection des répertoires (`<Directory>`)

On peut protéger les répertoires et sous répertoires contenus dans **`/var/www/`** par des stratégies de droits définies par des options entre les balises **`<Directory /chemin/Rep> </Directory>`**

Ces options, qui peuvent êtres cumulées, sont les suivantes :

- **`Indexes : .....`** Permet l'affichage du contenu d'un répertoire (en l'absence d'un fichier *index.html*)
- **`FollowSymLinks : .....`** Permet l'accès direct à une ressource par un script contenu dans une page (utile par exemple pour l'URL rewriting)
- **`Multiviews : .....`** Permet la sélection automatique et l'affichage de page dans une langue différente (en fonction de navigateur client)
- **`ExecCgi : .....`** Autorise l'exécution de scripts CGI

Les paramètres par défaut sont ceux de la racine.

De plus, la variable **`AllowOverride`** permet de spécifier ou non la détection d'un fichier *.htaccess* dans un répertoire, ce qui entraîne une authentification obligatoire. Lorsque sa valeur est « **`all`** », la vérification préalable de fichier *.htaccess* est activée. Si sa valeur est « **none** », la protection *HTACCESS* est désactivée. On spécifie ensuite l'ordre d'exécution de ces règles : **`Order allow,deny`** (ou l'inverse) et on spécifie enfin qui est autorisé et qui ne l'est pas.

On commence par épurer le site par défaut :

- Suppression de l'alias `/doc/` qui permet d'accéder à la documentation d'apache (qui est hébergée dans `/usr`)
- Configuration de l'adresse mail de l'administrateur du serveur (variable `ServerAdmin`)
- Suppression des options sur le répertoire `/var/www/` : plus de listage si le répertoire ne contient pas d'index, plus de suivi des liens symboliques etc.
- Suppression de l'argument `DAV on`

Ce qui donne :

```
<VirtualHost *:80>
      ServerAdmin postemaster@groupeirhenvironnement.eu
      DocumentRoot /var/www
      <Directory />
            Options FollowSymLinks
            AllowOverride None
      </Directory>
      <Directory /var/www/>
            AllowOverride None
            Order allow,deny
            allow from all
      </Directory>
      ScriptAlias /cgi-bin/ /usr/lib/cgi-bin/
      <Directory "/usr/lib/cgi-bin">
            AllowOverride None
            Options +ExecCGI -MultiViews +SymLinksIfOwnerMatch
            Order allow,deny
            Allow from all
      </Directory>
      # Emplacement des logs pour /var/www/
      ErrorLog ${APACHE_LOG_DIR}/error.log
```

```
      LogLevel warn
      CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

C. Création d'un nouveau site apache pour gérer notre serveur WebDAV :

On crée le site apache « webdav » dans `/etc/apache2/sites-available/` à partir du fichier « default »

- `cp /etc/apache2/sites-available/default /etc/apache2/sites-available/webdav`

En parallèle, créer sur votre serveur le répertoire « webdav » où seront stockés les fichiers « web » : nous le placerons dans `/var/www/webdav/`. Il faudra également donner les droits en lecture et écriture (car apache devra pouvoir écrire dans le répertoire afin que les utilisateurs puissent faire des dépôts de fichiers)

- `mkidr /var/www/webdav` : création du répertoire
- `chown www-data :www-data /var/www/webdav` : l'utilisateur exécutant le processus apache2 est déclaré utilisateur et groupe propriétaire
- `chmod 775 /var/www/webdav` : les propriétaires (utilisateur et groupe www-data) peuvent lire et écrire dans le répertoire, les autres n'ont que le droit de parcours et lecture.

```
root@webdav01:~# mkdir /var/www/webdav
root@webdav01:~# chown www-data:www-data /var/www/webdav/
root@webdav01:~# chmod 775 /var/www/webdav/
root@webdav01:~#
```

Il va maintenant falloir charger notre nouveau site « webdav » : créer le lien symbolique du site dans `/etc/apache2/sites-enabled`

Désactiver aussi le site par défaut : `rm /etc/apache2/sites-enabled/000-default`

⇨ **Pour prendre en compte les modifications, il faudra recharger apache…**

```
root@webdav01:/etc/apache2/sites-enabled# ln -s /etc/apache2/sites-available/webdav webdav
root@webdav01:/etc/apache2/sites-enabled# ll
total 0
lrwxrwxrwx 1 root root 26 14 janv. 13:35 000-default -> ../sites-available/default
lrwxrwxrwx 1 root root 35 14 janv. 18:54 webdav -> /etc/apache2/sites-available/webdav
root@webdav01:/etc/apache2/sites-enabled# invoke-rc.d apache2 reload
```

Testez, avec un navigateur, l'accès aux URL suivantes :

- http://@IP-de-votre-serveur/ => doit lister un répertoire vide
- http://nom-dns-de-votre-serveur/ => même chose que ci-dessus

Si vous testez de nouveau l'accès au répertoire avec un client WebDAV, vous constaterez que l'accès en écriture est maintenant permis ! Et oui, souvenez vous, `www-data` à le droit d'écrire dans le répertoire grâce à la commande **chown**.

D. Configuration de WebDAV

Pour le moment, votre serveur WebDAV est fonctionnel, mais en matière de sécurité, seuls les droits du système de fichiers régulent l'accès en lecture / écriture…
Apache va permettre de changer cela grâce à la mise en place d'une authentification et de la gestion des droits avec l'argument **LIMIT**.

Testons cela : nous allons créer un sous répertoire « *pot* » dans notre dossier WebDAV.
Pour ce répertoire, `www-data` aura le droit d'écriture. En parallèle, nous allons supprimer le droit d'écriture de notre racine.

```
root@webdav01:~# cd /var/www
root@webdav01:/var/www# chmod 555 webdav/
root@webdav01:/var/www# mkdir ./webdav/pot
root@webdav01:/var/www# chown www-data:www-data ./webdav/pot/
root@webdav01:/var/www# chmod 775 ./webdav/pot/
```

Au niveau des droits, nous avons maintenant ceci :

```
./webdav:
total 408
dr-xr-xr-x 5 www-data www-data   4096 15 janv. 13:38 .
drwxr-xr-x 3 root     root       4096 14 janv. 18:19 ..
-rw-r--r-- 1 www-data www-data 392703 14 janv. 19:26 Compte rendu sur la webconf _EQUALLOGIC_.pdf
drwxr-xr-x 2 www-data www-data   4096 14 janv. 19:26 .DAV
drwxrwxr-x 2 www-data www-data   4096 15 janv. 13:38 pot
drwxr-xr-x 2 www-data www-data   4096 14 janv. 19:29 test-rep
```

Notre racine n'est plus inscriptible via WebDAV, son sous répertoire pot, est lui inscriptible, sans authentification…

Via les droit « *directory* » d'apache, nous allons créer une authentification sur le répertoire « pot » pour limiter l'accès en écriture. Il faut éditer notre site apache « webdav »

Dans un premier temps, nous allons faire une *authentification* apache basique : on vérifie dans un fichier texte la présence d'un utilisateur et la validité du mot de passe qui lui est associé.

Nous allons placer ce fichier dans `/etc/apache2` et nous appellerons le fichier « *wdutils* ». Pour peupler (et créer le fichier s'il n'existe pas), nous utiliserons l'utilitaire *htpasswd* avec l'argument « –c »

Créons notre premier utilisateur, dans mon cas « toto » :

- `htpasswd –c /etc/apache2/wdutils toto`

```
root@webdav01:/var/www# htpasswd -c /etc/apache2/conf.d/wdutils toto
New password:
Re-type new password:
Adding password for user toto
```

Ne pas tenir compte de l'emplacement erroné du fichier sur la capture !

Si on regarde le fichier *wdutils*, il s'agit d'un fichier contenant la liste des utilisateurs et leurs mots de passe cryptés en **md5**

```
root@webdav01:/var/www# cat /etc/apache2/conf.d/wdutils
toto:3dpSQ/rjCX9GE
```

Sécurité : n'autoriser qu'apache (utilisateur et groupe) à lire le fichier uniquement

- `chmod 550 /etc/apache2/wdutils`

Maintenant, nous allons modifier le site apache webdav afin de gérer les droits sur notre nouveau répertoire « pot » pour lequel nous définirons un accès en écriture pour les utilisateurs présents dans notre fichier *wdutils* :

```
# Droits sur le repertoire pot
<Directory /var/www/webdav/pot/>
        # protection du repertoire par htaccess -> desactivee car on utilise les regle du directory
        AllowOverride none
        # on autorise le listage
        Options Indexes
         # Autorisation d'acces au site pour tout le monde
        Order allow,deny
        allow from all
        # Prise en charge de WebDAV pour ce rep
        DAV on

        # Parametres secu WebDAV
        # Interdiction des requetes PROFIND
        DavDepthInfinity off
        # Temps minimum de verrouillage d un fichier en edition (en sec)
        DavMinTimeout 120

        # Access avec authentification
        AuthName "Stockage WebDAV"
        # Activation de l'authentification apoache basic
        AuthType Basic
        # fichier contenant la liste des utilisateurs
        AuthUserFile /etc/apache2/conf.d/wdutils
        # droits accordes aux utilisateurs du fichier
        <Limit PUT POST DELETE PROPFIND PROPPATCH MKCOL COPY MOVE LOCK UNLOCK>
                Require valid-user
                # tout utilisateur present dans wdutils avec le bon mot de passe est autorise
        </Limit>
</Directory>
```

Ne pas tenir compte de l'emplacement erroné du fichier sur la capture !

Il faut recharger apache pour tester !

Note :

On peut appliquer plusieurs niveaux de sécurité : par exemple n'autoriser que l'utilisateur « jean » du fichier *wdutils* (avec « `require-user jean` ») ou encore, n'autoriser l'accès qu'à certaines adresses IP (ou même qu'à partir d'un seul réseau)… Il est aussi possible de « jouer » sur les droits définis par la balise « `<Limit> </Limit>` »

De même, nous pourrions créer un certificat auto-signé pour faire fonctionner notre site webdav en HTTPS seulement…

… ou encore faire une authentification basée sur un annuaire LDAP…

⇨ **Testez de nouveau l'accès !**

Juste pour exemple, je mets en place ici une nouvelle arborescence avec accès SSL **forcé** *et authentification LDAP :*

- Génération d'une clef privée et d'un *certificat X509* auto signé contenant la clef publique
- Ajout de la variable **REFERRALS off** dans `/etc/ldap/ldap.conf`
- Activation du module SSL
- Activation du module `authnz-ldap`
- Activation du module de réécriture d'URL (`rewrite`)
- Création d'un nouveau site SSLwebdav pour l'écoute d'apache sur le port 443
- Modification du site apache webdav (avec authentification LDAP et passage forcé en SSL)

```
root@webdav01:/var/www/webdav# ls -all
total 20
drwxrwxr-x 5 www-data www-data 4096 15 janv. 20:33 .
drwxr-xr-x 3 root     root     4096 14 janv. 18:19 ..
drwxr-xr-x 2 www-data www-data 4096 15 janv. 20:01 jbarbe
drwxr-xr-x 2 www-data www-data 4096 15 janv. 20:02 Public
drwxr-xr-x 2 www-data www-data 4096 15 janv. 20:02 sinfo
```

Nouvelle arborescence de ma racine WebDAV

Aperçu de mon site apache SSL :

```
<VirtualHost *:443>
      # Noms DNS de reponse du site apache
      # Si une demande URL est faite sur ces deux noms, c'est cette
configuration qui est appliquee
      ServerName webdav01.corp.lan
      ServerName webdav.corp.lan

      # Administrateur de ce site
      ServerAdmin jeremy.barbe@groupeirhenvironnement.eu

      # Emplacement des logs pour le site WebDAV
      # ATTENTION : le repertoire doit exister et etre accessible en
ecriture pour
      # l'utiilsateur www-data
      ErrorLog ${APACHE_LOG_DIR}/webdav/SSLerror.log
      LogLevel warn
      CustomLog ${APACHE_LOG_DIR}/webdav/SSLaccess.log combined

      # Prise en charge de SSL
      SSLEngine on
      SSLCertificateFile /etc/apache2/server.crt
      SSLCertificateKeyFile /etc/apache2/server.key
      # Emplacement du repertoire de base du site
      DocumentRoot /var/www/webdav/



      # Droits sur le repertoire racine
      <Directory /var/www/webdav/>
            AllowOverride None
            # Autorisation du listage du repertoire
            Options Indexes
            # Autorisation d'acces au site pour tout le monde
            Order allow,deny
            allow from all
            # Prise en charge de WebDAV pour ce rep
            DAV on
            # Gestion des requetes PROFIND
            DavDepthInfinity on
```

```
            # Temps minimum de verrouillage d un fichier en edition (en
sec)
            DavMinTimeout 10
            # Access restreint au personnel du groupe IRH
            AuthType Basic
            AuthBasicProvider ldap
            AuthzLDAPAuthoritative Off
            AuthName "Serveur WebDAV Groupe IRH"
            AuthLDAPURL
"ldap://@IP_serveur_LDAP/dc=corp,dc=lan?sAMAccountName"
            AuthLDAPBindDN "binduser@DOMAINE.DOM"
            AuthLDAPBindPassword "bindupass"
            Require valid-user
      </Directory>
      <Directory /var/www/webdav/jbarbe>
            Dav On
            AuthType Basic
            AuthBasicProvider ldap
            AuthzLDAPAuthoritative Off
            AuthName "Stockage personnel de Jérémy BARBÉ"
            AuthLDAPURL
"ldap://@IP_serveur_LDAP/dc=corp,dc=lan?sAMAccountName"
            AuthLDAPBindDN "binduser@DOMAINE.DOM"
            AuthLDAPBindPassword "bindupass"
            #Require valid-user
            Require user jbarbe
      </Directory>
      <Directory /var/www/webdav/sinfo>
            Dav On
            AuthType Basic
            AuthBasicProvider ldap
            AuthzLDAPAuthoritative Off
            AuthName "Stockage WebDAV service Informatique"
            AuthLDAPURL
"ldap://@IP_serveur_LDAP/dc=corp,dc=lan?sAMAccountName"
            AuthLDAPBindDN "binduser@DOMAINE.DOM"
            AuthLDAPBindPassword "bindupass"
            Require ldap-group CN=SinfoAdmin,CN=Users,DC=corp,DC=lan
      </Directory>
</VirtualHost>
```

E. Sécuriser apache avec Fail2Ban

Maintenant que notre service est fonctionnel, il ne reste plus qu'à activer sa protection avec Fail2Ban… Passer à **true** les sections relatives à apache et vérifier les actions menées en fonction des fichiers de configuration présents dans `/etc/fail2ban/filter.d/`

ATTENTION : *n'activez Fail2BAN qu'une fois votre serveur pleinement fonctionnel, sinon, en cas de trop nombreuses erreurs d'accès ou d'authentification, votre IP sera bannie…*

- ⇨ **Il ne reste plus qu'à appliquer la même installation et configuration d'apache / webdav sur votre second serveur ! Soyez rigoureux et pensez bien à modifier les éléments variables ! (ex : « *Servername* »)**
- ⇨ **SCP pourra vous être utile si vous ne souhaitez pas réécrire l'ensemble de vos fichiers de configuration…**

6. Mise en place du cluster Haute disponibilité

A. Notions Hearbeat / Pacemaker :

Maintenant que nos deux serveurs sont opérationnels avec une configuration rigoureusement identique, nous pouvons mettre en place un système de redondance.

Cela se passe en trois étapes :

- Bascule automatique de l'adressage IP entre les deux hôtes
- Mise en place d'un système RAID « réseau » pour le stockage WebDAV / Apache
- Synchronisation régulière des fichiers de configuration entre les deux hôtes

Heartbeat est un gestionnaire de cluster : il est en charge de la gestion des communications et de la gestion (ex : bascule d'un nœud en tant qu'esclave du cluster) de chaque nœud du cluster. Il est en charge de démarrer des scripts en fonction de l'état des nœuds qu'il supervise. (apache fait appel à des scripts de services présents dans `/etc/init.d/` ; mais peut aussi utiliser des scripts conçu par l'utilisateur)

Dans sa version *1*, HeartBeat ne supervisait que l'état global d'un nœud :

⇨ le serveur répond, il est en ligne, il ne répond pas, il est hors en ligne.

Depuis sa version 2, HeartBeat intègre un module supplémentaire : **pacemaker**. Pacemaker est un gestionnaire de ressources (ou CRM : *Content Resources Manager*). C'est un service capable de vérifier l'état d'un service (ex : apache) sur un nœud. Pour cela, pacemaker fait appel aux scripts d'état (*status*) de services présents dans `/etc/ini.d/`

Exemple pour apache : `/etc/init.d/ apache2 status` (même chose que `invoke-rc.d apache2 status`)

Ainsi, HeartBeat peut agir en fonction de l'état d'un nœud (éteint, injoignable) mais aussi en fonction de l'état des services qu'il doit rendre (dans le cas d'un service planté, HeartBeat va essayer de le redémarrer et en cas d'un certain nombre d'échecs, choisira de basculer l'ensemble des services à rendre hautement disponibles sur un autre nœud)

HearBeat peut utiliser plusieurs types de média pour la communication « inter-nœuds » :

- Un câble série (RS232) : difficile dans un environnement virtuel
- Le réseau Ethernet : un câble croisé entre deux nœuds physiques, du broadcast, du mulicast ou encore de l'unicast.

Dans l'état actuel, nous avons donc deux nœuds : *webdav01 et webdav02* avec une seule carte réseau : celle qui est sur le réseau de production et qui utilisée pour répondre aux requêtes du service apache.

Nous allons modifier légèrement notre infrastructure : nous allons relier nos deux hôtes via un nouveau réseau IP qui servira dans un premier temps aux communications d'HeartBeat. Il sera ensuite utilisé pour la réplication des données en temps réél via DRBD. Ce nouveau réseau sera

complètement dédié à cela et, en conséquence, ne devra pouvoir recevoir que deux hôtes (sécurité supplémentaire). Nous connecterons nos deux hôtes à ce réseau en utilisant une nouvelle carte réseau et le lien entre nos deux hôtes sera en gigabit…

Nous définissons donc le réseau « cluster » : 192.168.10.0/30 (30 bits = 2 octets disponibles pour la partie machine - « host » - du réseau).
Ce réseau met donc à notre disposition 4 adresses IP, dont deux réservées (une pour l'adresse réseau, une pour le broadcast) : soit 2 adresses machines.

- Adresse réseau : 192.168.10.0
- 1ere IP : 192.168.10.1
- 2ème IP : 192.168.10.2
- Adresse de broadcast : 192.168.10.3

Ainsi, sur chaque nœud :

- Ajouter une carte « eth1 » (modifier le fichier `/etc/network/interfaces`)
- Configurer l'adresse IP permettant de joindre le réseau 192.168.10.0/30
- Remplacer le nom machine : *dav-node1* et *dav-node2* (remplacent respectivement *webdav01* et *webdav02* qui seront alors de simples entrées DNS utilisées pour le service apache)
- Modifier le fichier `/etc/hosts` de façon à permettre à chaque nœud de résoudre les noms « *dav-node1* » et « *dav-node2* » qui seront utilisés uniquement par les services de haute disponibilité

Exemple de configuration réseau sur le premier hôte « *dav-node1* » (anciennement « *webdav01* ») :

```
# This file describes the network interfaces available on your syste
# and how to activate them. For more information, see interfaces(5).

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
allow-hotplug eth0
iface eth0 inet static
address 172.20.160.13
netmask 255.255.240.0
gateway 172.20.160.3

# Interface ETH1 pour la replication DRBD
allow-hotplug eth1
iface eth1 inet static
address 192.168.10.1
netmask 255.255.255.252

~
~
~
"/etc/network/interfaces" 20L, 475C                           1,1
```

Fichier */etc/network/interfaces/*

```
dav-node1
~
```

Fichier `/etc/hostname`

```
127.0.0.1       localhost
172.20.160.13   webdav01.corp.lan       webdav01

# Resolution des noms pour DRBD
192.168.10.1    dav-node1
192.168.10.2    dav-node2

# The following lines are desirable for IPv6 capable hosts
::1     ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
~
```

Fichier `/etc/hosts`

Pensez également à modifier le fichier de configuration de **Fail2BAN** afin d'ignorer les communications entre les deux nœuds de notre futur cluster qui utiliseront cette nouvelle carte réseau :

```
# The DEFAULT allows a global definition of the options. They can be override
# in each jail afterwards.

[DEFAULT]

# "ignoreip" can be an IP address, a CIDR mask or a DNS host
ignoreip = 127.0.0.1

# Adresse IP d'un poste d'administration
ignoreip = 172.20.160.54
#adresses IP des hotes + IP virtuelle du cluster HA
ignoreip = 172.20.160.13
ignoreip = 172.20.160.14
ignoreip = 172.20.160.15
ignoreip = 192.168.10.1
ignoreip = 192.168.10.2

# Duree de l'interdiction de dialogue en secondes
bantime  = 600 #10 min

# Nombres maximum d'echecs de connexion sur le serveur
maxretry = 3
"/etc/fail2ban/jail.conf" 299L, 7050C Ã©crit(s)              36,1          4%
```

Fichier `/etc/fail2ban/jail.conf`

Dans le fichier de configuration apache (`/etc/apache2/apache2.conf`), ajouter la variable `ServerName webdav.corp.lan` (futur nom DNS pour les requêtes apache) pour éviter le message d'erreur indiquant qu'aucun nom d'hôte n'est défini pour le fonctionnement d'apache

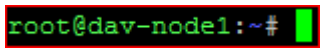
```
Servername webdav.corp.lan
"/etc/apache2/apache2.conf" 232L, 8022C Ã©crit(s)
```

Redémarrer vos deux hôtes. Vérifier que :

- `Uname –n` doit maintenant retourner le nouveau nom DNS (ex : *dav-node1*)
- Vous pouvez « pinger » entre eux vos noeuds sur chaque interface
- Chaque nœud doit pouvoir résoudre le nom DNS « interne » du cluster et le nom DNS dédié à apache (ex : *dav-node1* et *webdav01*)

- Que les paquets passent par la bonne interface réseau (le but étant qu'à terme, DRDB n'utilise que ETH1, qu'apache ne travaille que sur ETH0 et que HearBeat travaille sur les 2 … pour la sécurité)

A noter : au prompt, vous devriez *avoir root@dav-node**x*** et non plus *root@webdav0**x***...




```
root@dav-node1:~# ping dav-node2
PING dav-node2 (192.168.10.2) 56(84) bytes of data.
64 bytes from dav-node2 (192.168.10.2): icmp_req=1 ttl=64 time=0.155 ms
^C
--- dav-node2 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.155/0.155/0.155/0.000 ms
root@dav-node1:~# ping webdav02
PING webdav02.corp.lan (172.20.160.14) 56(84) bytes of data.
64 bytes from webdav02.corp.lan (172.20.160.14): icmp_req=1 ttl=64 time=0.147 ms
64 bytes from webdav02.corp.lan (172.20.160.14): icmp_req=2 ttl=64 time=0.132 ms
^C
--- webdav02.corp.lan ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 999ms
rtt min/avg/max/mdev = 0.132/0.139/0.147/0.014 ms
root@dav-node1:~# uname -n
dav-node1
root@dav-node1:~# arp -a
webdav02.corp.lan (172.20.160.14) at 00:50:56:aa:35:15 [ether] on eth0
dav-node2 (192.168.10.2) at 00:50:56:aa:26:c1 [ether] on eth1
vostro19.corp.lan (172.20.160.58) at 00:25:64:e7:ba:99 [ether] on eth1
vostro19.corp.lan (172.20.160.58) at 00:25:64:e7:ba:99 [ether] on eth0
meldes.corp.lan (172.20.160.7) at 00:50:56:87:05:90 [ether] on eth0
root@dav-node1:~#
```

Série de vérifications des modifications réseau

Bascule des ressources entre les nœuds avec HeartBeat / Pacemaker :

Principe : nos deux hôtes ont pour le moment deux adresses IP et deux noms DNS différents. Nous allons utiliser une nouvelle adresse IP associée à un nouveau nom DNS. Le serveur opérationnel devra répondre sous cette IP et ce nom DNS. En cas de défaillance de l'hôte opérationnel (par défaut webdav01) Heartbeat basculera cette nouvelle adresse IP sur l'hôte de secours (webdav02)

Heartbeat devra être installé sur les deux hôtes et configuré de façon strictement identique. Nous installerons HearBeat depuis les sources de paquets Debian. La version 2 sera automatiquement installée avec ces dépendances (Pacemaker, inclus par défaut avec la version 6 – *Squeeze* – de Debian sera également mis à niveau)

Avant de commencer l'installation et la configuration d'HeartBeat / Pacemaker, il y'a trois termes que vous entendrez souvent lorsque l'on parle d'un cluster : Il s'agit de 3 problématiques liées au concept de « cluster » :

- Le « **split-brain** » peut intervenir quand chaque nœud de votre cluster croit son voisin hors service. Il va alors prendre la main : chaque nœud est donc maître et fournit le service. Les effets de bords peuvent être gênants dès lors qu'on utilise une ressource partagée.

Pour se protéger un maximum de ce problème, il est fortement recommandé d'utiliser deux liens réseau différents pour la communication entre les nœuds (ou de faire du *bonding*).

Un autre problème possible : quand votre nœud maître rencontre un problème, le cluster « bascule » . Un autre nœud prend la main et devient maître à son tour : il fournit le service. Mais dans quel état est l'autre nœud ? Comment s'assurer qu'il ne fournit pas encore le service au risque de bloquer/corrompre des ressources partagées ? Il n'y a pas vraiment de réponse à ce problème mais une solution, pas forcement évidente à mettre en place : **Stonith**.

- **Stonith ou « Shoot The Other Node In The Head** » ou encore « *Shoot The Offending Node In The Head* ». C'est une méthode d'isolation d'un nœud (*fencing*) qui pour une raison ou une autre ne répond plus. L'idée est d'éviter à tout prix le fameux **split-brain** qui peut amener tous vos nœuds à fournir le même service (ou à ne plus le fournir du tout). Le nœud jugé hors service sera donc, en général, redémarré ou arrêté électriquement (via IPMI par exemple).

Pour finir, un autre point bloquant peut être le **quorum**.

- **Le Quorum** : c'est le nombre minimum de nœuds en ligne requis pour être capable de valider une décision. Dans le cas d'un cluster avec Pacemaker, il faut que plus de la moitié des nœuds soit en ligne. Avec un cluster à deux nœuds, il n'y a plus de quorum dès qu'un nœud est perdu. Il va donc falloir demander à Pacemaker d'ignorer le quorum dans cette situation. Le fonctionnement par défaut quand le quorum n'est pas atteint est de couper toutes les ressources !

B. Installation d'Heartbeat

```
apt-get install heartbeat
```

Note :

Répertoire des fichiers de configuration d'hearbeat : `/etc/heartbeat/` (il s'agit d'un alias pointant sur le répertoire `/etc/ha.d/`)

Documentation : `/usr/share/doc/heartbeat`

- Par défaut, aucun fichier de configuration n'est présent. Des modèles sont disponibles avec la documentation.
- On pourra mettre en place Heartbeat à partir de ces fichiers modèles.
- Heartbeat est un service : à chaque modification de sa configuration, il est nécessaire de relire les fichiers de configuration (`invoke rc.d heartbeat reload`)

Pacemaker travaille, quant à lui, avec des fichiers XML, son fichier de configuration est `/var/lib/heartbeat/crm/cib.xml`

C. Configuration d'Heartbeat / Pacemaker :

- `/etc/heartbeat/ha.cf` : c'est le fichier de configuration général d'HeartBeat. Il sert à paramétrer la façon dont les nœuds vont communiquer (choix du ou des médias à utiliser pour les battements de cœur), l'intervalle de vérification de l'état des nœuds, le délais au bout duquel les hôtes sont déclarés « mort » et la journalisation des événements dans les logs…
- `/etc/heartbeat/haresources` : c'est le fichier dans lequel on indique quelles ressources doivent être rendues hautement disponibles. Il s'agit des ressources pour lesquelles HeartBeat doit être capable de lancer un script d'arrêt/marche de la ressource sur chacun des nœuds (exemple : lancer `/etc/init.d/apache2 start` sur le nœud 1 pour démarrer apache)
- `/etc/heartbeat/authkeys` : ce fichier permet de sécuriser la communication entre chaque nœud du cluster. Le plus simple dans un réseau inter-nœud « fiable » est d'utiliser une passphrase…

Rappel : ne pas confondre *Heartbeat* et *Pacemaker*. Heartbeat ne saura jamais dans quel état est un service. C'est le rôle de Pacemaker. Heartbeat sait juste dans quel état est chaque nœud du clutser qu'il gère et sait démarrer un service sur un nœud en faisant appel aux scripts présents dans `/etc/init.d/`

ATTENTION : tous ces fichiers devront être systématiquement identiques sur chaque nœud du cluster ! Pensez toujours à répercuter vos modifications sur l'ensemble des nœuds (SCP vous fera gagner du temps !)… Comme d'habitude, après chaque modification, il faudra faire un petit `invoke-rc.d heartbeat reload`

Dans la version 1 d'HeartBeat, il était impératif d'installer un gestionnaire de ressources (Pacemaker, CoroSync etc.) Dans la version 2, Pacemaker est intégré, il suffit de l'activer en ajoutant la variable `crm yes` dans `/etc/heartbeat/ha.cf`

Le fichier `haresources` est simple : il est généralement composé d'une seule ligne contenant toutes les ressources du cluster. Il est composé d'appels de scripts (ex : `IPAddr` pour le passage d'une adresse IP) et de paramètres qui sont passés à des scripts par `::`

Le fichier est lu de gauche à droite dans le cas d'une bascule des ressources du nœud maître vers l'esclave et inversement dans le cas d'une bascule du nœud esclave vers le nœud maître. **Il est donc important de ne pas se tromper dans l'ordre des services !** En effet, si vous démarrez apache avant d'avoir monté le volume ou se trouvent les données d'apache, il risque d'y avoir quelque petits problèmes…

Dans un premier temps, nous allons créer une configuration de base. Cette dernière devra assurer, en cas de perte du nœud 1 que les services soient démarrés sur le nœud 2. On parle ici d'une surveillance globale de l'état des nœuds, on ne s'occupe pas encore de leurs services. Il s'agit d'une configuration sans Pacemaker…

⇨ Créer, **sur chaque nœud** les fichiers suivants :

Fichier ha.cf (`/etc/heartbeat/ha.cf`)

```
# Fichier log de débug
debugfile /var/log/ha-debug

# Fichier log
logfile /var/log/ha-log

# Temps en secondes entre chaque battement de cœur
keepalive       2

# Au bout de 10s sans battement de cœur, la machine est considérée comme « morte »
# Si c'est le maître, les services seront basculés vers l'esclave.
deadtime        10

# Au bout de 7s sans battement de coeur
# une alerte sera générée dans les logs
warntime            7

# Interface(s) utilisée(s)  pour les battements de cœur.
# On utilisera nos deux interfaces réseau ETH0 et ETH1 (on limite le « split brain »).
#  Plusieurs modes de dialogue sont possibles : le brodacast, le multicast et l'unicast.
# Je préfère ici l'utilisation de l'unicast : les battements de cœurs ne sont échangé qu'entre ces IP,
# cela évite de saturer le réseau…
# ATTENTION : Impossible de faire de l'ajout de nœud supplémentaire en auto via cette conf
ucast eth0 172.20.160.13
ucast eth0 172.20.160.14

# Utilisation en secours des ETH1 dédies au réseau DRBD
ucast eth1 192.168.10.1
ucast eth1 192.168.10.2

#Lors du crash du maître, les ressources sont basculées vers l'esclave.
#Ce paramètre mis sur "on" demande de renvoyer les ressources sur le maitre dès son retour.
#Sur "off" les ressources reste sur l'esclave même si le maître est de nouveau opérationnel.
#PS: Sur off, dans le cas où l'esclave tomberait en panne mais que le maitre est en standby, celui-ci
reprendrait les ressources.
auto_failback    on

# Augmentation de 90s du deadtime en cas de redémarrage de la machine
initdead            60

# Port UDP par défaut pour communiquer les battements de cœur.
udpport             694

# Lien par le port série pour communiquer les battements de cœur => Désactivé !!
#serial /dev/ttyS0

# Le nom de chaque serveur obtenu avec uname -n
node      dav-node1
node      dav-node2

# Gestion des ressources (services via Pacemaker, uniquement pour Heartbeat2)
crm no # Pas de pacemaker dans un premier temps…
```

Fichier authkeys (`/etc/heartbeat/authkeys`)

```
#     Authentication file.  Must be mode 600
#
#       Must have exactly one auth directive at the front.
#       auth    send authentication using this method-id
#
#       Then, list the method and key that go with that method-id
#
#       Available methods: crc sha1, md5.  Crc doesn't need/want a key.
#
#       You normally only have one authentication method-id listed in this
file
#
#       Put more than one to make a smooth transition when changing auth
#       methods and/or keys.
#
#       sha1 is believed to be the "best", md5 next best.
#       crc adds no security, except from packet corruption.
#               Use only on physically secure networks.
#
auth 1
1 sha1 WebDAVHA!
```

Je ne commenterai pas plus le fichier : la communication entre les nœuds est chiffrée en MD5 avec la passphrase « *WebDAVHA !* »

Pour que tout fonctionne, il est nécessaire de pouvoir exécuter le fichier (et au minimum et que les autres ne puissent pas le lire… sécurité !)

- `Chmod 600 /etc/heartbeat/authkeys`

Fichier haresources (`/etc/heartbeat/haresources`)

```
# Attribution de l'IP 172.20.160.15/20 (webdav.corp.lan) à l'hôte actif
# et démarrage d'apache2
#
#dav-node1 : serveur opérationnel en condition normale (le maître)
# 172.20.160.15/20/eth0 =  attribuer l'IP 172.20.160.15 avec le masque
# 255.255.240.0 sur la carte eth0
# apache2 =  démarrer apache sur l'hôte en service
dav-node1 IPaddr::172.20.160.15/20/eth0 apache2
```

… et c'est tout !! Enfin presque ! Heartbeat va donc se charger d'attribuer l'IP 172.20.160.15 à l'hôte maître des opérations (ou à l'esclave en cas de bascule…) et va démarrer apache… Afin d'éviter tout problème, il vaudrait mieux qu'apache ne démarre plus en automatique au démarrage des serveurs… Pour cela, désactivez le démarrage d'apache :

- `update-rc.d apache2 disable`

Testons : démarrez Heartbeat sur chaque serveur (après avoir stoppé apache via `/etc/init.d/apache2 stop`) : `invoke-rc.d heartbeat start`

Faire un ping de l'IP que Heartbeat doit attribuer… Elle devrait répondre… De même, si vous appelez cette IP dans un navigateur, vous devriez accéder à votre serveur WebDAV !

Vérifions ce qu'il s'est passé : `invoke-rc.d heartbeat status` devrait vous retourner le nœud maître des opérations :

```
root@dav-node1:/etc/heartbeat# invoke-rc.d heartbeat status
heartbeat OK [pid 1613 et al] is running on dav-node1 [dav-node1]...
root@dav-node1:/etc/heartbeat#
```

Le nœud dav-node1 est maître : il exécute les ressources demandées

Puisque *dav-node1* est maître, apache doit être actif sur ce dernier… vérifions :

```
root@dav-node1:/etc/heartbeat# invoke-rc.d apache2 status
Apache2 is running (pid 10710).
root@dav-node1:/etc/heartbeat# ps -aux | grep "apache2"
Warning: bad ps syntax, perhaps a bogus '-'? See http://procps.sf.net/faq.html
root     10710  0.0  1.7  38452  9148 ?        Ss   15:16   0:00 /usr/sbin/apache2 -k start
www-data 10729  0.0  1.0  38932  5476 ?        S    15:16   0:00 /usr/sbin/apache2 -k start
www-data 10730  0.0  1.1  39008  6096 ?        S    15:16   0:00 /usr/sbin/apache2 -k start
www-data 10731  0.0  1.2  38712  6408 ?        S    15:16   0:00 /usr/sbin/apache2 -k start
www-data 10732  0.0  1.2  38736  6520 ?        S    15:16   0:00 /usr/sbin/apache2 -k start
www-data 10734  0.0  0.8  38452  4624 ?        S    15:16   0:00 /usr/sbin/apache2 -k start
www-data 15259  0.0  0.8  38452  4612 ?        S    15:41   0:00 /usr/sbin/apache2 -k start
www-data 15324  0.0  0.8  38452  4612 ?        S    15:41   0:00 /usr/sbin/apache2 -k start
root     19894  0.0  0.1   3324   876 pts/0    S+   19:05   0:00 grep --color=auto apache2
root@dav-node1:/etc/heartbeat#
```

Deux façons de vérifier l'état d'un service

Enfin, puisque *dav-node1* est maître, il devrait avoir une adresse IP supplémentaire : la 172.20.60.15

```
root@dav-node1:/etc/heartbeat# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:50:56:aa:0d:54
          inet adr:172.20.160.13  Bcast:172.20.175.255  Masque:255.255.240.0
          adr inet6: fe80::250:56ff:feaa:d54/64 Scope:Lien
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:523454 errors:0 dropped:0 overruns:0 frame:0
          TX packets:143375 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 lg file transmission:1000
          RX bytes:52389734 (49.9 MiB)  TX bytes:40210134 (38.3 MiB)

eth0:0    Link encap:Ethernet  HWaddr 00:50:56:aa:0d:54
          inet adr:172.20.160.15  Bcast:172.20.175.255  Masque:255.255.240.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1

eth1      Link encap:Ethernet  HWaddr 00:50:56:aa:2b:d0
          inet adr:192.168.10.1  Bcast:192.168.10.3  Masque:255.255.255.252
          adr inet6: fe80::250:56ff:feaa:2bd0/64 Scope:Lien
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:519530 errors:0 dropped:0 overruns:0 frame:0
          TX packets:141110 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 lg file transmission:1000
          RX bytes:50014725 (47.6 MiB)  TX bytes:25253424 (24.0 MiB)

lo        Link encap:Boucle locale
          inet adr:127.0.0.1  Masque:255.0.0.0
          adr inet6: ::1/128 Scope:HÃ´te
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:294902 errors:0 dropped:0 overruns:0 frame:0
          TX packets:294902 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 lg file transmission:0
          RX bytes:71559060 (68.2 MiB)  TX bytes:71559060 (68.2 MiB)
```

Enfin, vous devez pouvoir faire la vérification **inverse** sur le second nœud qui doit être en mode esclave :

```
root@dav-node2:~# invoke-rc.d apache2 status
Apache2 is NOT running.
invoke-rc.d: initscript apache2, action "status" failed
root@dav-node2:~# ifconfig | grep "172.20.160.15"
root@dav-node2:~#
```

Apache n'est pas en exécution et l'IP n'est pas attribuée

Tout semble fonctionner, **testons une bascule** : éteignez votre nœud 1… Rapidement, vous devriez pouvoir continuer à accéder à votre site WebDAV car le nœud 2 devrait avoir pris le relais…

- vérifier que votre nœud secondaire s'est bien vu attribué votre IP virtuelle (ou VIP)
- Vérifier qu'apache est en exécution sur le nœud secondaire

Redémarrez votre nœud primaire, la bascule inverse doit se faire…

Maintenant que la base est prête, ajoutons Pacemaker à notre système de haute disponibilité : le but étant de surveiller en plus l'état du service à rendre hautement disponible.

Activer Pacemaker : éditer le fichier `/etc/heartbeat/ha.cf` et passer `crm` à `yes`. **A faire sur les deux nœuds. Relancer heartbeat sur chaque nœud…**

```
# Gestion des ressources (services via Pacemaker, uniquement pour Heartbeat2)
crm yes
"ha.cf" 46L, 1521C Ã©crit(s)                                                46,
```

Pacemaker fonctionne avec un fichier XML. Il va falloir le générer via la commande suivante : `python /usr/lib/heartbeat/haresources2cib.py`

- Cela entraîne la création du fichier `/var/lib/heartbeat/crm/cib.xml`
- Passer la commande sur les deux nœuds et relancer heartbeat…
- Pacemaker dispose d'un outil de monitoring que l'on peut appeler via `crm_mon`

`crm_mon` retourne le statut du cluster :

```
============
Last updated: Sun Jan 22 19:44:54 2012
Stack: Heartbeat
Current DC: dav-node2 (d6a4223f-1871-43b7-bb9e-fa353c9c2d9f) - partition with quorum
Version: 1.0.9-74392a28b7f31d7ddc86689598bd23114f58978b
2 Nodes configured, unknown expected votes
1 Resources configured.
============

Online: [ dav-node1 dav-node2 ]


Failed actions:
    IPaddr_172_20_160_15_start_0 (node=dav-node2, call=4, rc=1, status=complete): unknown erro
r
    IPaddr_172_20_160_15_start_0 (node=dav-node1, call=4, rc=1, status=complete): unknown erro
r
```

Si vous coupez votre nœud maître, vous allez constater que la bascule ne se fait pas ! Il y'a deux raisons à cela : une erreur dans la création du fichier `cib.xml` et un problème de quorum !

Un petit coup d'œil au fichier cib.xml va vous donner une piste !

```
<instance_attributes id="IPaddr_172_20_160_15_inst_attr">
  <attributes>
    <nvpair id="IPaddr_172_20_160_15_attr_0" name="ip" value="172.20.160.15"/>
    <nvpair id="IPaddr_172_20_160_15_attr_1" name="cidr_netmask" value="eth0"/>
    <nvpair id="IPaddr_172_20_160_15_attr_2" name="nic" value="20"/>
  </attributes>
</instance_attributes>
```

- La valeur du masque en notation CIDR est « eth0 » alors qu'elle devrait être « 20 » !
- La valeur de la carte réseau où appliquer l'adresse virtuelle devrait être ETH0 et non « 20 » !

Que s'est il passé ? Et bien, le script Python qui crée le fichier CIB.XML à partir de `haresources` contient un bug… ça arrive !

Si on regarde un peu plus haut dans le fichier, un autre problème :

```
<nvpair id="cib-bootstrap-options-symmetric-cluster" name="symmetric-cluster" value="true"/>
<nvpair id="cib-bootstrap-options-no-quorum-policy" name="no-quorum-policy" value="stop"/>
```

`no-quorum-policy` à pour valeur « stop »… Qu'est ce que cela veut dire ? Et bien, si on perd le quorum, alors les services devront être stoppés ! Et cela n'est pas compatible dans un cluster où il n'y a que deux nœuds…

Pour rappel : un quorum, dans le cadre d'un cluster, c'est prendre des décisions en fonction d'un nombre de votes (ce sont les nœuds qui votent !). Le quorum implique que pour valider une action, il faut avoir une majorité absolue dans les votes (soit la moitié du nombre des nœuds du cluster +1) Si il n'y a que deux nœuds, le quorum est 2… cela n'est donc pas possible…

- Il faut alors modifier la valeur « stop » par « **ignore** » : on ignore le quorum dans notre cluster de deux nœuds…

Corrigeons tout cela :
Il y'a deux méthodes : la lourde mais simple, la légère mais plus complexe… Je ne détaillerai que la méthode lourde, étant celle que j'ai utilisé… (je conseille, dans le cas où vous choisiriez **crm** pour configurer pacemaker de créer complètement le fichier via l'utilitaire…)

Commençons par la méthode lourde, à faire sur chaque nœud :

- Sur chaque nœud, couper heartbeat : `invoke-rc.d heartbeat stop`
- Supprimer tous les fichiers présents dans `/var/lib/heartbeat/crm/`
- Régénérer le fichier `cib.xml : python /usr/lib/heartbeat/haresources2cib.py`
- Corriger les erreurs et ignorer le quorum
- Démarrer Heartbeat

Avec la méthode « légère, on utilisera l'utilitaire crm :

Pour configurer votre cluster, nous allons nous connecter au cli Pacemaker (sur le nœud de votre choix : on pourra pousser la config sur l'ensemble des nœuds via `commit`) via la commande *crm* :

```
root@dav-node1:/var/lib/heartbeat/crm# crm
crm(live)# help

This is the CRM command line interface program.

Available commands:

        cib             manage shadow CIBs
        resource        resources management
        node            nodes management
        options         user preferences
        configure       CRM cluster configuration
        ra              resource agents information center
        status          show cluster status
        quit,bye,exit   exit the program
        help            show help
        end,cd,up       go back one level

crm(live)#
```

```
crm(live)configure# property stonith-enabled="false" no-quorum-policy="ignore"
crm(live)configure# show
node $id="d6a4223f-1871-43b7-bb9e-fa353c9c2d9f" dav-node2
node $id="f7ecb4b6-f347-49c3-8131-59d4bbcb458a" dav-node1
property $id="cib-bootstrap-options" \
        dc-version="1.0.9-74392a28b7f31d7ddc86689598bd23114f58978b" \
        cluster-infrastructure="Heartbeat" \
        stonith-enabled="false" \
        no-quorum-policy="ignore"
```

On désactive le stonith et on ignore le quorum…

Une fois les modifications nécessaires effectuées, il ne reste plus qu'à vérifier : lancer `crm_mon` :

```
============
Last updated: Sun Jan 22 20:58:24 2012
Stack: Heartbeat
Current DC: dav-node2 (d6a4223f-1871-43b7-bb9e-fa353c9c2d9f) - partition with quorum
Version: 1.0.9-74392a28b7f31d7ddc86689598bd23114f58978b
2 Nodes configured, unknown expected votes
1 Resources configured.
============

Online: [ dav-node1 dav-node2 ]

 Resource Group: group_1
     IPaddr_172_20_160_15       (ocf::heartbeat:IPaddr):        Started dav-node1
     apache2_2  (lsb:apache2):  Started dav-node1
```

Voila qui est mieux !

Faisons un test de bascule en arrêtant le premier nœud : éxécutez `crm_mon` sur le deuxième nœud, vous devriez constater que ce dernier est devenu maître et qu'il exécute les ressources (vérifiez le avec un navigateur…)

```
Attempting connection to the cluster...Your configuration was internally updated
 to the latest version (pacemaker-1.0)
============
Last updated: Sun Jan 22 21:01:32 2012
orumk: Heartbeat
Version: 1.0.9-74392a28b7f31d7ddc86689598bd23114f58978b2d9f) - partition with qu
2 Nodes configured, unknown expected votes
1 Resources configured.
============

Online: [ dav-node2 ]
OFFLINE: [ dav-node1 ]

 Resource Group: group_1
     IPaddr_172_20_160_15       (ocf::heartbeat:IPaddr):        Started dav-node
2
     apache2_2  (lsb:apache2):  Started dav-node2
```

Le premier nœud est hors ligne, l'esclave a donc pris le relais !

Si vous redémarrez le premier nœud, les services doivent rebasculer sur le primaire (comme défini dans `ha.cf` avec l'argument `auto_failback on`)

```
============
Last updated: Sun Jan 22 21:08:00 2012
Stack: Heartbeat
Current DC: dav-node2 (d6a4223f-1871-43b7-bb9e-fa353c9c2d9f) - partition with qu
orum
Version: 1.0.9-74392a28b7f31d7ddc86689598bd23114f58978b
2 Nodes configured, unknown expected votes
1 Resources configured.
============

Online: [ dav-node1 dav-node2 ]

 Resource Group: group_1
     IPaddr_172_20_160_15       (ocf::heartbeat:IPaddr):        Started dav-node
1
     apache2_2  (lsb:apache2):  Started dav-node1
```

Les services ont basculé sur le nœud maître dès son retour…

Il ne nous reste plus qu'à valider l'intérêt majeur de Pacemaker : que se passe t-il si un service critique (comprendre un service que l'on doit surveiller) vient à planter ?
Dans le cas d'apache, Pacemaker va surveiller son état : s'il est arrêté, Pacemaker va le relancer. Au bout de 3 échecs, Pacemaker va demander à HeartBeat de basculer les ressources sur le second nœud…

Testons : via la commande `kill`, tuer le processus apache !

```
root@dav-node1:~# ps -aux | grep "apache2"
Warning: bad ps syntax, perhaps a bogus '-'? See http://procps.sf.net/faq.html
root      1929  0.0  1.7  38452  9144 ?        Ss   21:08   0:00 /usr/sbin/apache2 -k start
www-data  1932  0.0  0.8  38452  4624 ?        S    21:08   0:00 /usr/sbin/apache2 -k start
www-data  1933  0.0  0.8  38452  4612 ?        S    21:08   0:00 /usr/sbin/apache2 -k start
www-data  1934  0.0  0.8  38452  4612 ?        S    21:08   0:00 /usr/sbin/apache2 -k start
www-data  1935  0.0  0.8  38452  4612 ?        S    21:08   0:00 /usr/sbin/apache2 -k start
www-data  1936  0.0  0.8  38452  4612 ?        S    21:08   0:00 /usr/sbin/apache2 -k start
root      3103  0.0  0.1   3320   812 pts/0    S+   21:14   0:00 grep --color=auto apache2
root@dav-node1:~# kill 1929
root@dav-node1:~# invoke-rc.d apache2 status
Apache2 is NOT running.
invoke-rc.d: initscript apache2, action "status" failed.
```

Au bout de quelques secondes, apache 2 est redémarré :

```
root@dav-node1:~# invoke-rc.d apache2 status
Apache2 is running (pid 3344).
```

… et les services sont toujours sur le même nœud :

```
Online: [ dav-node1 dav-node2 ]

 Resource Group: group_1
     IPaddr_172_20_160_15       (ocf::heartbeat:IPaddr):        Started dav-node
1
     apache2_2  (lsb:apache2):  Started dav-node1
```

Testons maintenant avec une « attaque » contre apache qui devrait l'empêcher de redémarrer :

- renommer le fichier de configuration `apache2.conf` en `apache2.conf.BACK`
- tuer le processus apache

```
Apache2 is running (pid 3344).
root@dav-node1:~# mv /etc/apache2/apache2.conf /etc/apache2/apache2.conf.BACK
root@dav-node1:~# kill 3344
root@dav-node1:~#
```

- Pacemaker va tenter de redémarrer le service, après échecs, il demandera à Heartbeat de basculer les ressources sur le nœud secondaire :

```
Online: [ dav-node1 dav-node2 ]

 Resource Group: group_1
     IPaddr_172_20_160_15       (ocf::heartbeat:IPaddr):        Started dav-node2
     apache2_2  (lsb:apache2):  Started dav-node2
```

Il faudra attendre un petit peu, le temps que pacemaker tente le redémarrage d'apache et fasse basculer les ressources.

⇨ N'oubliez pas de remettre votre fichier de configuration en état… Il faudra ensuite redémarrer apache à la main (avec la commande `/etc/init.d/apache2 start`) afin que la bascule vers le nœud primaire ait lieue.

Voila pour la partie gestion de la haute disponibilité des nœuds et services.

Il reste maintenant un problème de taille : notre serveur WebDAV va stocker les fichiers dans l'emplacement défini dans son fichier de configuration de site… Problème, cet emplacement de stockage est local à chaque nœud : en cas de bascule, on ne retrouve pas ses données !

Comment faire ?

- Syncrhoniser les fichiers via Rsync : oui, mais les données ne seront pas disponibles en temps réél…
- Utiliser une baie SAN : trop coûteux
- Un partage NFS ? Il faudra configurer tout cela … mais ça pourrait aller
- DRBD !

Nous utiliserons la 4ème option : **DRBD**

7. Synchronisation des données en temps réél avec DRBD

A. Notions

DRBD (*Distributed Replicated Block Device* en anglais, ou périphérique en mode bloc répliqué et distribué en français) est une architecture de stockage distribuée pour GNU/Linux, permettant la réplication de périphériques de bloc (disques, partitions, volumes logiques etc…) entre des serveurs. DRBD est un logiciel libre, mais un support existe. Ce support est réalisé par la société Linbit. DRBD est composé d'un module noyau, d'outils d'administration en espace utilisateur ainsi que de scripts shell.

La réplication des données se fait:

- En temps réel. En permanence, pendant que les applications modifient les données présentes sur le périphérique
- De façon transparente. Les applications qui stockent leur données sur le périphérique répliqué n'ont pas conscience que ces données sont en fait stockées sur plusieurs ordinateurs
- De façon synchrone, ou asynchrone. En fonctionnement synchrone, une application qui déclenche une écriture de donnée est notifiée de la fin de l'opération seulement après que l'écriture a été effectuée sur tous les serveurs, alors qu'en fonctionnement asynchrone, la notification se fait après que la donnée a été écrite localement, mais avant la propagation de la donnée.

Principe de fonctionnement :

DRBD ajoute une couche logique de périphériques de bloc (conventionnellement nommée `/dev/drbdX`, où X est le numéro de périphérique mineur) au dessus de la couche logique locale des périphériques de bloc existante sur les nœuds du cluster participants. Les écritures sur le nœud primaire sont transférées sur le périphérique de bloc de bas niveau et sont simultanément propagées au nœud secondaire. Le nœud secondaire transfère ensuite les données à son périphérique de bloc de bas niveau correspondant. Toutes les lectures sont effectuées localement.

Source : http://fr.wikipedia.org/wiki/DRBD

B. Mise en place et configuration d'un nouveau disque dur dédié à DRBD

Maintenant que les présentations sont faites, attaquons le vif du sujet. Nous allons, sur chaque nœud ajouter un nouveau disque dur sur lequel sera créée une nouvelle partition. Cette partition locale sera synchronisée en temps réel avec la partition logique de DRBD qui elle-même sera synchronisée avec la partition DRBD (et ainsi la partition logique locale) du second nœud.

Sur chacun de vos serveurs, ajouter un nouveau disque (de taille identique ou non, le principal étant que la partition du nœud primaire puisse loger dans le disque ajouté sur le nœud secondaire)

Pour information : Nom des périphériques sous Linux

Le nom des disques et des partitions sous Linux diffère des autres systèmes d'exploitation. Vous devez connaître les noms utilisés lors du partitionnement.

Voici les conventions de nommage :

- Le premier lecteur de disquette est nommé « /dev/fd0 ».
- Le second lecteur de disquette est nommé « /dev/fd1 ».
- Le premier disque SCSI (selon l'identifiant SCSI) est nommé « /dev/sda ».
- Le second disque SCSI (selon l'identifiant) est nommé « /dev/sdb », ainsi de suite.
- Le premier CD-ROM SCSI est nommé « /dev/scd0 », ou encore « /dev/sr0 ».
- Le disque maître sur le contrôleur IDE primaire est nommé « /dev/hda ».
- Le disque esclave sur le contrôleur IDE primaire est nommé « /dev/hdb ».
- Les disques maître et esclave sur le second contrôleur sont nommés respectivement « /dev/hdc » et « /dev/hdd ». Les nouveaux contrôleurs IDE peuvent avoir deux canaux fonctionnant comme deux contrôleurs distincts. Les lettres peuvent différer de ce qui apparaît dans le programme mac pdisk (ce qui apparaît comme /dev/hdc dans pdisk peut apparaître comme /dev/hda avec Debian).

Les partitions sur chaque disque sont représentées en ajoutant un numéro au nom du disque : « sda1 » et « sda2 » représentent la première et la seconde partition du premier disque SCSI du système.

Voici un exemple concret : Supposons que vous ayez deux disques SCSI, l'un à l'adresse SCSI 2 et l'autre à l'adresse 4. Le premier disque (à l'adresse 2) est nommé « sda », et le second « sdb ». Si le disque « sda » a 5 partitions, elles s'appelleront « sda1 », « sda2 », ..., « sda5 ». La même convention s'applique au disque « sdb » et ses partitions.

Une fois le disque dur installé (dans le cas d'une machine virtuelle, simplement ajouté à la VM), il convient de créer une partition sur ce dernier.

Les opérations qui suivent devront être effectuées sur chaque nœud :
Nous utiliserons fdisk.

`Fdisk -l` permettra d'afficher les disques installés et les partitions présentes.
Vous obtiendrez quelque chose comme cela :

```
Disk /dev/sda: 32.2 GB, 32212254720 bytes
255 heads, 63 sectors/track, 3916 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x00029d5c

   Device Boot      Start         End      Blocks   Id  System
/dev/sda1   *           1        3793    30461952   83  Linux
/dev/sda2            3793        3917      992257    5  Extended
/dev/sda5               3793          3917       992256    82   Linux swap /
Solaris

Disk /dev/sdb: 32.2 GB, 32212254720 bytes
255 heads, 63 sectors/track, 3916 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
```

```
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x00000000

Disk /dev/sdb doesn't contain a valid partition table
```

- Le nouveau disque est le périphérique `/dev/sdb`, il n'y à aucune partition sur ce dernier
- Nous allons donc créer une nouvelle partition sur le disque **en appelant fdisk sur le disque** `/dev/sdb`

```
root@server1:~# fdisk /dev/sdb
Device contains neither a valid DOS partition table, nor Sun, SGI or OSF
disklabel
Building a new DOS disklabel with disk identifier 0x8042e800.
Changes will remain in memory only, until you decide to write them.
After that, of course, the previous content won't be recoverable.

Warning: invalid flag 0x0000 of partition table 4 will be corrected by
w(rite)

WARNING: DOS-compatible mode is deprecated. It's strongly recommended to
         switch off the mode (command 'c') and change display units to
         sectors (command 'u').

Command (m for help): <-- n pour créér une nvle partition
Command action
   e   extended
   p   primary partition (1-4)
<-- p pour créer une partition primaire
Partition number (1-4): <-- 1 : on créé la 1ere partition
First cylinder (1-3916, default 1): <-- Presser ENTREE
Using default value 1
Last cylinder, +cylinders or +size{K,M,G} (1-3916, default 3916): <-- ENTREE
Using default value 3916

Command (m for help): <-- t : on définit le type de la partition
Selected partition 1
Hex code (type L to list codes): <-- 83 , le type 83 correspond à ext3

Command (m for help): <-- w , permet d'écrire la table des partitions
The partition table has been altered!

Calling ioctl() to re-read partition table.
Syncing disks.
root@server1:~#
```

Vérification avec `fdisk -l` :

```
Disk /dev/sdb: 8589 MB, 8589934592 bytes
255 heads, 63 sectors/track, 1044 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0xdf3223fc

   Device Boot      Start         End      Blocks   Id  System
/dev/sdb1               1        1044     8385898+  83  Linux
```

Notre partition est prête !

C. Mise en place de DRBD et installation des utilitaires :

DRBD est présent dans le noyau de debian depuis la version 5. Il n'est donc plus nécessaire de le recompiler à partir des sources. Il suffit d'activer DRBD avec la commande : `modprobe drbd`

Pour vérifier, utiliser la commande : `lsmod | grep « drbd »`

```
root@dav-node1:/var/lib/heartbeat/crm# lsmod | grep "drbd"
drbd                  173348  3
lru_cache               4054  1 drbd
cn                      3667  1 drbd
```

Vous devriez obtenir quelque chose comme cela….

Il ne reste plus qu'à installer les utilitaires DRBD : `apt-get install drbd8-utils`

D. Configuration de DRBD :

Le fichier principal de configuration de DRBD est `/etc/drbd.conf`. Par défaut, il fait appel à des sous fichiers de configuration qui se situent dans `/etc/drbd.conf`

- nous n'utiliserons pas ces fichiers mais placerons tout dans le fichier `drbd.conf`

Supprimer le contenu du fichier `drbd.conf` et le remplacer par ce qui suit :

```
# Des exemples de fichiers de conf dans :
/usr/share/doc/drbd.../drbd.conf.example

# JAR, 15/01/12 : suppression des appels vers les fichiers de /etc/drbd.d/
#include "drbd.d/global_common.conf";
#include "drbd.d/*.res";
# => toute la config est chargée depuis ce fichier unique

global { usage-count no; }
common { syncer { rate 100M; } }
# Création de la ressource "r0"
# ... il pourrait très bien y en avoir plusieurs... Je l'ai appelée r0 mais
le nom n'a pas d'importance !
resource r0 {
        protocol C;
# Trois protocoles sont disponibles :
# En protocole A, l'acquittement d'écriture (sur le maître)
# est envoyé dès que les données ont été transmises au
# sous volume physique et envoyées à l'esclave.
# En protocole B, l'acquittement d'écriture (sur le maître)
# est envoyé dès que les données ont été transmises au
# sous volume physique et reçues par l'esclave.
# En protocole C, l'acquittement d'écriture (sur le maître)
# est envoyé dès que les données ont été transmises au
# sous volume physique ET au sous volume physique de
# l'esclave.
# Le C est donc le plus sûr ; vu la qualité de notre liaison
#(lien giga dédié) on peut l'utiliser sans problème.

        startup {
                wfc-timeout  15;
# Au démarrage du noeud, on attend (cela bloque le
```

```
# démarrage) le noeud distant pendant 15 secondes.
                degr-wfc-timeout 60;
# Au démarrage d'un noeud précédemment dégradé on
# attend (cela bloque le démarrage) le noeud
# distant pendant 1 minute.
        }
        net {
                cram-hmac-alg sha1;
                shared-secret "webdavHA";
# On sécurise le dialogue entre les noeuds avec du MD5 et une passphrase
        }
# Declaration du premier noeud : on utilise son nom DNS, celui résolu par
uname -n
        on dav-node1 {
                device /dev/drbd0;
# volume logique créé par DRBD
                disk /dev/sdb1;
# Volume logique local qui sera synchronisé avec le volume DRBD
                address 192.168.10.1:7788;
# Adresse IP du noeud utilisée pour la réplication DRBD vers l'autre noeud
:
# j'utilise ici l'IP associée à ETH1, interface réseau GB sur le réseau
dédié à DRBD
                meta-disk internal;
# emplacement des meta-data.
        }
# Déclaration du second noeud
        on dav-node2 {
                device /dev/drbd0;
                disk /dev/sdb1;
                address 192.168.10.2:7788;
                meta-disk internal;
        }
}
```

Sur chaque nœud, initialiser votre volume DRBD *(/dev/drbd*) via la commande `drbdadm create-md r0` : on créé ainsi le volume DRBD de la ressource r0 définie précédemment

```
root@dav-node1:~# drbdadm create-md r0
Writing meta data...
initializing activity log
NOT initialized bitmap
New drbd meta data block successfully created.
```

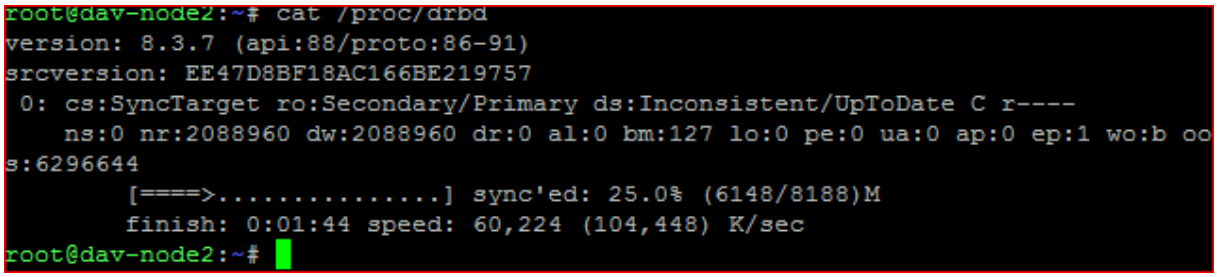
Démarrer DRBD sur chaque nœud : `invoke-rc.d drbd start`

Sur le nœud maître de votre cluster (ici, *dav-node1*), initialiser la synchronisation entre votre volume DRBD et votre partition locale. Ceci va : définir le nœud 1 comme maître au niveau de DRBD et synchroniser les données présentes dans `/dev/sdb1` sur `/dev/drbd0/`.

Les données seront synchronisées en temps réel sur /dev/drbd0 ainsi que sur /dev/sdb1 du second noeud.

- `drbdadm -- --overwrite-data-of-peer primary all`

Vous pouvez suivre l'avancement en temps réel de l'initialisation avec la commande `watch /proc/drbd (CTRL+C pour sortir)`



`cat /proc/drbd` *fonctionne aussi mais pas de temps réel !*

Au passage, *Secondary/primary* indique qui est maître au niveau de DRBD : ici je suis sur *dav-node2*, son rôle correspond à ce qui suit `ro :` => il est donc esclave

Une fois la synchronisation terminée, il ne reste plus qu'à créer un système de fichier sur notre partition DRBD. Sur le nœud maître au niveau de DRBD (c'est-à-dire le « primary) , exécuter la commande suivante :

- `mkfs.ext3 /dev/drbd0`

Tout est maintenant prêt ! Il n'y a plus qu'à tester ! Créer le répertoire `/data` via la commande

- `mkdir` /data

Montez ensuite votre partition DRBD dans ce répertoire (toujours depuis le premier nœud) :

- `mount /dev/drbd0 /data`
- Céér un fichier « toto.txt » `: touch /data/toto.txt`

Nous allons maintenant vérifier la présence de ce fichier sur le nœud « secondaire » ou « esclave ». Pour cela, il faut :

Sur le nœud primaire (« primary » au sens DRBD) :

- démonter notre partition sur le nœud primaire : `umount /data`
- basculer le nœud primaire en tant que secondaire au niveau de drbd : `drbdadm secondary r0`

Sur le noeud secondaire (« secondary » au sens DRBD) :

- créer le répertoire `/data` qui sera le point de montage de notre partition DRBD
- déclarer le nœud en tant que maître au niveau de DRBD : `drbdadm primary r0`
- Monter la partition dans `/data : mount /dev/drbd0 /data`

- `ls /data` doit vous afficher le fichier toto.txt

De la même façon, créé le fichier toto2 .txt, démontez le volume, repassez le nœud en secondaire. Sur l'autre nœud, le passer en « *primary* » au niveau de DRBD, monter la partition dans `/data`. Vous devez maintenant voir vos deux fichiers.

Commandes utiles à retenir :

- Afficher l'état du service DRBD : `invoke-rc.d drbd status`
- Afficher l'état des ressources DRBD : `cat /proc/drbd`
- Passer un nœud en maître : `drbdam primary r0`
- Passer un nœud en secondaire : `drbdadm secondary r0`

Forcer la resynchronisation entre les 2 serveurs après une vérification rapportant des anomalies

- `drbdadm disconnect r0`
- `drbdadm connect r0`

Supprimer les données du cluster de l'un des serveurs (à exécuter sur le serveur sur lequel les données sont à supprimer)

- `drbdadm disconnect r0`
- `drbdadm -- --discard-my-data connect r0`

Mise à jour de la configuration pour que le service prenne en compte une mise à jour du fichier de configuration. Ne pas oublier que la configuration doit être identique sur les 2 serveurs.

Pour prendre en compte toutes les modifications, quel que soient la ressource et aussi les modifications présentes dans la section commune:

- `drbdadm adjust all`

Pour ne prendre en compte que les modifications apportées à une ressource:

- `drbdadm adjust r0`

Pour résumer : nous savons donc gérer notre cluster avec Hearbeat, surveiller les services avec pacemaker, synchroniser nos données avec DRBD. Il nous reste à appliquer tout cela sur notre cluster !

8. Intégration de DRBBD au cluster Heartbeat

Nous souhaitons donc rendre hautement disponible notre serveur WebDAV. Dans le cas d'une bascule sur le nœud de secours, il faut donc que les données soient à jour ! Dans la configuration de Apache notre serveur WebDAV (fichier site apache *webdav*), nous avions défini l'emplacement de racine à `/var/www/webdav`

Deux options s'offrent à nous :

- modifier la configuration de notre site apache pour dire que les données sont par exemple dans `/data/webdav` (où `/data` est le point de montage de notre volume DRBD)
- monter directement notre volume DRBD dans `/var/www/`

J'ai choisi la 2ème option pour ne pas avoir à trop modifier ma configuration Apache/WebDAV…

Avant toute chose, commençons par copier les données existantes dans `/var/www/` dans notre partition DRBD (`/dev/drbd0`). Tout devra se faire depuis le nœud 1, maître de notre cluster.

- `drbdadm primary r0`
- `mount /dev/drbd0 /data`
- `cp –Rpv /var/www/* /data`

⇨ Toutes les données sont copiées (j'ai choisi de conserver les paramètres propriétaires et droits pendant la copie grâce à l'option **–p)**

Maintenant, supprimons le contenu de `/var/www/` :

⇨ `rm –rf /var/www/*`

Il ne nous reste plus qu'à monter notre volume DRBD dans /var/www/ :

⇨ `umount /data`
⇨ `mount /dev/drbd /var/www/`

Modifications à effectuer au niveau de WebDAV : au niveau de WebDAV, en cas de bascule, si un fichier est ouvert par un utilisateur, il faut maintenir son statut « lecture seule » lors de la bascule. WebDAV / Apache gèrent les fichiers accédés grâce à des « verrous ». Le verrou WebDAV (DavLockDB) est par défaut situé dans `/var/locks/apache2/`… il n'est donc pas synchrone sur les deux nœuds.

Pour corriger ce problème, il suffit de le déplacer dans `/var/www/`

- Créer le fichier de vérrou : `touch /var/www/DavLock`
- Rendre apache propriétaire : `chown www-data :www-data /var/www/DavLock`
- Attribuer les bons droits : `chmod 770 /var/www/DavLock`
- Modifier l'emplacement de ce fichier dans la configuration d'apache : pour ce faire, éditer le fichier `/etc/apache2/mods-available/dav_fs.conf` comme il suit (à faire sur les deux nœuds) :

```
#DAVLockDB ${APACHE_LOCK_DIR}/DAVLock
DAVLockDB /var/www/DavLock
```

Pour terminer, il faut modifier notre configuration Hearbeat de façon à permettre le montage de notre partition DRBD dans `/var/www/` au démarrage du nœud maître et sur l'esclave en cas de bascule !

Pour cela, rien de plus simple : on va ajouter une ressource dans notre fichier `/etc/heartbeat/haressources`

```
dav-node1 IPaddr::172.20.160.15/20/eth0 drbddisk::r0
Filesystem::/dev/drbd0::/var/www::ext3 apache2
```

```
dav-node1 IPaddr::172.20.160.15/20/eth0 drbddisk::r0 Filesystem::/dev/drbd0::/var/www::ext3 apache2
```

Au final, Heartbeat est en charge de passer l'adresse **IP 172.20.160.15/20** sur la carte **ETH0** du nœud actif puis de monter la partition **/dev/drbd0/** dans **/var/www/** et enfin de démarrer **apache2**

NOTE : le script *drbd* prend également en charge le fait de passer un nœud en *primary/secondary*… Par défaut, au démarrage des deux nœuds, ils sont tous en *secondary ;* heartbeat passera le nœud opérationnel en primary.

Afin de prendre en compte ces changements, sur chaque nœud :

- ⇨ Arrêter heartbeat : `invoke-rc.d heartbeat stop`
- ⇨ Supprimer le contenu du répertoire `/var/lib/heartbeat/crm/`

Sur le nœud primaire :

- ⇨ Modifier le fichier `/etc/heartbeat/haresources` comme plus haut
- ⇨ Générer le fichier `cib.xml` avec le script python : `python /usr/lib/heartbeat/haresources2cib.py`
- ⇨ Modifier le fichier **cib.xml** : corriger l'inversion `cid_netmask` et `inet` et passer le `no-quorum-policy` à « *ignore* »
- ⇨ Copier (via SCP) le fichier `/etc/haresources` et le fichier `/var/lib/heartbeat/crm/cib.xml` sur votre second nœud

- ⇨ **Assurez-vous que les fichiers de configuration d'apache et de sites apache sont identiques sur les deux nœuds.**

- ⇨ **Il ne vous reste plus qu'à redémarrer vos deux nœuds. C'est terminé, votre cluster devrait être complètement opérationnel !**

9. Conseils de maintenance :

En cas de maintenance ou de mise à jour nécessaire de votre cluster, le plus simple est de :

- Mettre hors ligne un des deux hôtes du cluster via CRM
- Arrêter le service Heartbeat sur les deux hôtes (si des redémarrages sont à prévoir, pensez à désactiver le service Heartbeat au démarrage des machines)
- Procéder à la maintenance / mise à jour sur l'hôte en ligne
- Tester / valider la maintenance
- Relancer Heartbeat sur chaque nœud et inverser les hôtes en ligne / hors ligne
- Procéder à la même maintenance / mise à jour sur le second nœud
- Au besoin, réactiver le démarrage d'Heartbeat au démarrage des machine et rebooter l'ensemble (ne pas oublier de sortir votre nœud du mode « hors ligne »)

10. Liens & références

A. *WebDAV :*

- http://www.pervasive-network.org/SPIP/Installation-de-WebDAV-pour

B. *HeartBeat / Pacemaker :*

- http://wapiti.telecom-lille1.eu/commun/ens/peda/options/ST/RIO/pub/exposes/exposesrio2007/legrand-playez/howto.htm
- http://binbash.fr/2011/09/19/des-clusters-avec-pacemaker/
- http://www.netexpertise.eu/fr/linux/heartbeat-2-mise-en-oeuvre.html
- http://www.octetmalin.net/linux/tutoriels/heartbeat-creer-cluster-serveurs-gestion-de-services-haute-disponibilite.php
- http://www.deimos.fr/blocnotesinfo/index.php?title=Installation_et_configuration_d%27un_cluster_Heartbeat_V2

C. *DRBD :*

- http://www.howtoforge.com/setting-up-network-raid1-with-drbd-on-debian-squeeze
- http://www.crium.univ-metz.fr/docs/system/drbd/