By Marc
Published: 2008-10-09 20:17

The Perfect Load-Balanced & High-Availability Web Cluster With 2 Servers Running Xen On Ubuntu 8.04 Hardy Heron Introduction

This is a copy paste from my site: **blogama.org**

In this howto we will build a load-balanced and high-availability web cluster on 2 real servers with Xen, hearbeat and ldirectord. The cluster will do http, mail, DNS, MySQL database and will be completely monitored. This is currently used on a production server with a couple of websites.

The goal of this tutorial is to achieve load balancing & high availability with as few real servers as possible and of course, with open-source software. More servers means more hardware & hosting cost.

Most of the information you will find here has been copy / pasted from a dozen howtos, many of them from **howtoforge.com**, but some important details have been modified to make this possible and to put everything together.

Here is a quick list of services & applications that will be installed:

- Apache
- MySQL + phpmyadmin
- Postfix (SMTP) with web based users configuration and Spamassassin
- Courier (IMAP & POP) and squirrelmail
- Bind (DNS server)
- Munin and monit for web based monitoring
- Homemade scripts for monitoring

What you need

2 servers with dual lan, at least 7 IPs. IPs will be used like this :

- dom01.example.com : 192.168.1.100
- dom02.example.com : 192.168.1.101
- lb1.example.com : 192.168.1.102
- lb2.example.com : 192.168.1.103
- web1.example.com : 192.168.1.104
- web2.example.com : 192.168.1.105
- example.com : 192.168.1.106
- yousite.com (optional) : 192.168.1.107

Dom0 will be separated from load balancers and web servers. I didn't try it but I believe it would be possible to put load balancers on Dom0.

I suggest at least 2GB ram and RAID 1 or 10 hard drives for a production server.

Limitations

1) This worked for me. Doesn't mean it will work for you but rest assured that the howto is 100% tested to work on a production and test server !

2) This setup is scalable over 2 servers but you will need to find another way for MySQL replication if you do so.

3) No control panel such as ISPConfig, CPanel, etc...

4) Some websites can break MySQL Master to Master replication. It happend to me with Drupal but I fixed it either by disabling cache or by setting a minimum cache lifetime. Please read this before you go further :

A: MySQL replication currently does not support any locking protocol between master and slave to guarantee the atomicity of a distributed (cross-server) update. In other words, it is possible for client A to make an update to co-master 1, and in the meantime, before it propagates to co-master 2, client B could make an update to co-master 2 that makes the update of client A work differently than it did on co-master 1. Thus, when the update of client A makes it to co-master 2, it produces tables that are different from what you have on co-master 1, even after all the updates from co-master 2 have also propagated. This means that you should not chain two servers together in a two-way replication relationship unless you are sure that your updates can safely happen in any order, or unless you take care of mis-ordered updates somehow in the client code.

1. Installing Ubuntu

Do a basic install of Ubuntu 8.04 LTS server edition.

If you want to install with software RAID 1 please read this howto I wrote :

Install Ubuntu 8.04 with software raid 1

2. Installing Xen

You can run Xen from image files or from dedicated partition. Both have pros and cons.

From image files disk I/O is slower but its easier to do backups and to manage. Its the other way around when working on a partition.

What I suggest doing is starting with image file and to end with partition when your setup is finished. This way you can do backups of your image files and rollback if necessary when testing.

To install on image files please refer to this great tutorial from the howto master Falko : **Installing Xen On An Ubuntu 8.04 (Hardy Heron) Server From The Ubuntu Repositories**

To install directly on a partition (my modified version of Falko's howto) : **High Performance XEN On An Ubuntu Hardy Heron (8.04) Server System AMD64 or i386**

You need to do 2 Xen domain on each server (dom01 and dom02 are Dom0 or VM controller) :

server #1 - dom01.example.com

lb1.example.com (256MB RAM - 5GB HD is enough)
ip : 192.168.1.102

web1.example.com (the more RAM the better, keep 512MB for Dom0)
ip : 192.168.1.104

server #2 - dom02.example.com

lb2.example.com (256MB RAM - 5GB HD is enough)
ip : 192.168.1.103

web2.example.com (the more RAM the better, keep 512MB for Dom0)
ip : 192.168.1.105

3. Creating Xen Bridges for local data transfers (optional)

By default only one network card is enabled on virtual machine with Xen. For local transfer such as rsync, MySQL replication and backups I use a gigabit crossover cable between the 2 servers. Its not necessary but it will savebandwidth costs and replication will be faster.

Please refer to this howto to create xen bridge : **Creating new xen bridges on Ubuntu 8.04**

In this howto ip used on the second network card (crossover) will be the following :

- dom01.example.com : 192.168.0.100
- dom02.example.com : 192.168.0.101
- lb1.example.com : 192.168.0.102
- lb2.example.com : 192.168.0.103
- web1.example.com : 192.168.0.104
- web2.example.com : 192.168.0.105

4. Node preparation (dom01, dom02, lb1, lb2, web1, web2)4.1 Installing openssh server and VIM

Run :

```
sudo su
apt-get install vim ssh openssh-server
```

4.2 Updating the repositories

```
mv /etc/apt/sources.list /etc/apt/sources.list.bak

vi /etc/apt/sources.list
```

Make sources.list look like this :

```
#
# deb cdrom:[Ubuntu-Server 8.04 _Hardy Heron_ - Release i386 (20080423.2)]/ hardy main restricted
#deb cdrom:[Ubuntu-Server 8.04 _Hardy Heron_ - Release i386 (20080423.2)]/ hardy main restricted
# See http://help.ubuntu.com/community/UpgradeNotes for how to upgrade to
# newer versions of the distribution.
deb http://de.archive.ubuntu.com/ubuntu/ hardy main restricted
deb-src http://de.archive.ubuntu.com/ubuntu/ hardy main restricted
## Major bug fix updates produced after the final release of the
## distribution.
deb http://de.archive.ubuntu.com/ubuntu/ hardy-updates main restricted
deb-src http://de.archive.ubuntu.com/ubuntu/ hardy-updates main restricted
## N.B. software from this repository is ENTIRELY UNSUPPORTED by the Ubuntu
## team, and may not be under a free licence. Please satisfy yourself as to
## your rights to use the software. Also, please note that software in
## universe WILL NOT receive any review or updates from the Ubuntu security
## team.
deb http://de.archive.ubuntu.com/ubuntu/ hardy universe
deb-src http://de.archive.ubuntu.com/ubuntu/ hardy universe
deb http://de.archive.ubuntu.com/ubuntu/ hardy-updates universe
deb-src http://de.archive.ubuntu.com/ubuntu/ hardy-updates universe
## N.B. software from this repository is ENTIRELY UNSUPPORTED by the Ubuntu
## team, and may not be under a free licence. Please satisfy yourself as to
## your rights to use the software. Also, please note that software in
```

```
## multiverse WILL NOT receive any review or updates from the Ubuntu
## security team.
deb http://de.archive.ubuntu.com/ubuntu/ hardy multiverse
deb-src http://de.archive.ubuntu.com/ubuntu/ hardy multiverse
deb http://de.archive.ubuntu.com/ubuntu/ hardy-updates multiverse
deb-src http://de.archive.ubuntu.com/ubuntu/ hardy-updates multiverse
## Uncomment the following two lines to add software from the 'backports'
## repository.
## N.B. software from this repository may not have been tested as
## extensively as that contained in the main release, although it includes
## newer versions of some applications which may provide useful features.
## Also, please note that software in backports WILL NOT receive any review
## or updates from the Ubuntu security team.
# deb http://de.archive.ubuntu.com/ubuntu/ hardy-backports main restricted universe multiverse
# deb-src http://de.archive.ubuntu.com/ubuntu/ hardy-backports main restricted universe multiverse
## Uncomment the following two lines to add software from Canonical's
## 'partner' repository. This software is not part of Ubuntu, but is
## offered by Canonical and the respective vendors as a service to Ubuntu
## users.
# deb http://archive.canonical.com/ubuntu hardy partner
# deb-src http://archive.canonical.com/ubuntu hardy partner
deb http://security.ubuntu.com/ubuntu hardy-security main restricted
deb-src http://security.ubuntu.com/ubuntu hardy-security main restricted
deb http://security.ubuntu.com/ubuntu hardy-security universe
deb-src http://security.ubuntu.com/ubuntu hardy-security universe
deb http://security.ubuntu.com/ubuntu hardy-security multiverse
deb-src http://security.ubuntu.com/ubuntu hardy-security multiverse
```

Now do :

```
apt-get update
```

```
apt-get upgrade
```

4.3 Modifications

`/bin/sh` is a symlink to `/bin/dash`, however we need `/bin/bash`, not `/bin/dash`. Therefore we do this:

```
ln -sf /bin/bash /bin/sh
```

We will disable AppArmor (on dom01 and dom02) by doing the following :

```
/etc/init.d/apparmor stop

update-rc.d -f apparmor remove
```

5. Network configuration (dom01, dom02, lb1, lb2, web1, web2)

5.1 Setting up IPs

To edit network configuration under Ubuntu do :

```
vi /etc/network/interfaces
```

We will now do each network configuration one by one. I assume you use 2 network card, eth0 is the one connected to the internet and eth1 the one with

the crossover cable. I wont write the config file individually, only for dom01.example.com, please modify accordingly to this list :

dom01.example.com

eth0 : 192.168.1.100
eth1 : 192.168.0.100

dom02.example.com

eth0 : 192.168.1.101
eth1 : 192.168.0.101

lb1.example.com

eth0 : 192.168.1.102
eth1 : 192.168.0.102

lb2.example.com

eth0 : 192.168.1.103
eth1 : 192.168.0.103

web1.example.com

eth0 : 192.168.1.104
eth1 : 192.168.0.104

web2.example.com

eth0 : 192.168.1.105
eth1 : 192.168.0.105

Example network configuration of dom01.example.com :

Make the file `/etc/network/interfaces` look like this :

```
# 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 connected to the internet
auto eth0
iface eth0 inet static
	address 192.168.1.100
	netmask 255.255.255.0
	network 192.168.1.0
	broadcast 192.168.1.255
	gateway 192.168.1.1
# The secondary network interface connected by a crossover cable on the other server
auto eth1
iface eth1 inet static
	address 192.168.0.100
	netmask 255.255.255.0
	network 192.168.0.0
	broadcast 192.168.0.255
```

Now save the file and do :

```
/etc/init.d/networking restart
```

5.2 Hostname

```
vi /etc/hosts
```

and make it look like this, otherwise you will have problems with ldirectord later on :

dom01.example.com

```
127.0.0.1       localhost.localdomain   localhost
127.0.1.1       dom01.example.com dom01
192.168.1.101   dom02.example.com     dom02
192.168.1.102   lb1.example.com     lb1
192.168.1.103   lb2.example.com     lb2
192.168.1.104   web1.example.com     web1
192.168.1.105   web2.example.com     web2
# 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
ff02::3 ip6-allhosts
```

```
echo dom01.example.com > /etc/hostname

/etc/init.d/hostname.sh start
```

dom02.example.com

```
127.0.0.1       localhost.localdomain   localhost
127.0.1.1       dom02.example.com dom02
192.168.1.100   dom01.example.com     dom01
192.168.1.102   lb1.example.com     lb1
192.168.1.103   lb2.example.com     lb2
192.168.1.104   web1.example.com     web1
192.168.1.105   web2.example.com     web2
# 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
ff02::3 ip6-allhosts
```

```
echo dom02.example.com > /etc/hostname

/etc/init.d/hostname.sh start
```

lb1.example.com

```
127.0.0.1       localhost.localdomain   localhost
127.0.1.1       lb1.example.com lb1
192.168.1.100   dom01.example.com     dom01
192.168.1.101   dom02.example.com     dom02
192.168.1.103   lb2.example.com     lb2
192.168.1.104   web1.example.com     web1
192.168.1.105   web2.example.com     web2
# 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
ff02::3 ip6-allhosts
```

```
echo lb1.example.com > /etc/hostname

/etc/init.d/hostname.sh start
```

lb2.example.com

```
127.0.0.1       localhost.localdomain   localhost
127.0.1.1       lb2.example.com lb2
192.168.1.100   dom01.example.com     dom01
192.168.1.101   dom02.example.com     dom02
192.168.1.102   lb1.example.com     lb1
192.168.1.104   web1.example.com     web1
192.168.1.105   web2.example.com     web2
# 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
ff02::3 ip6-allhosts
```

```
echo lb2.example.com > /etc/hostname

/etc/init.d/hostname.sh start
```

web1.example.com

```
127.0.0.1       localhost.localdomain   localhost
127.0.1.1       web1.example.com web1
192.168.1.100   dom01.example.com     dom01
192.168.1.101   dom02.example.com     dom02
192.168.1.102   lb1.example.com     lb1
192.168.1.103   lb2.example.com     lb2
192.168.1.105   web2.example.com     web2
# 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
ff02::3 ip6-allhosts
```

```
echo web1.example.com > /etc/hostname

/etc/init.d/hostname.sh start
```

web2.example.com

```
127.0.0.1       localhost.localdomain   localhost
127.0.1.1       web2.example.com web2
192.168.1.100   dom01.example.com     dom01
192.168.1.101   dom02.example.com     dom02
192.168.1.102   lb1.example.com     lb1
192.168.1.103   lb2.example.com     lb2
192.168.1.104   web1.example.com     web1
# 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
ff02::3 ip6-allhosts
```

```
echo web2.example.com > /etc/hostname

/etc/init.d/hostname.sh start
```

6. Software installation (dom01, dom02, lb1, lb2, web1, web2)

Run :

```
apt-get install binutils cpp fetchmail flex gcc libarchive-zip-perl libc6-dev libcompress-zlib-perl libdb4.3-dev libpcre3 libpopt-dev lynx m4 make ncftp nmap openssl perl perl-modules unzip zip zlib1g-dev autoconf automake1.9 libtool bison autotools-dev g++ build-essential
```

7. Apache/PHP5/Ruby (web1, web2)7.1 Software installation

Now we install Apache:

```
apt-get install apache2 apache2-doc apache2-mpm-prefork apache2-utils libexpat1 ssl-cert
```

Next we install PHP5 and Ruby (both as Apache modules):

```
apt-get install libapache2-mod-php5 libapache2-mod-ruby php5 php5-common php5-curl php5-dev php5-gd php5-idn php-pear php5-imagick php5-imap
php5-mcrypt php5-memcache php5-mhash php5-ming php5-mysql php5-pspell php5-recode php5-snmp php5-sqlite php5-tidy php5-xmlrpc php5-xsl
```

7.2 Apache configuration

Next we edit `/etc/apache2/mods-available/dir.conf` :

```
vi /etc/apache2/mods-available/dir.conf
```

and change the DirectoryIndex line:

```
<IfModule mod_dir.c>

        #DirectoryIndex index.html index.cgi index.pl index.php index.xhtml index.htm
        DirectoryIndex index.html index.htm index.shtml index.cgi index.php index.php3 index.pl index.xhtml
</IfModule>
```

Now we have to enable some Apache modules (SSL, rewrite, suexec, and include):

```
a2enmod ssl

a2enmod rewrite

a2enmod suexec

a2enmod include
```

Reload the Apache configuration:

```
/etc/init.d/apache2 force-reload
```

We will add some domains in the vhost configuration file :

```
mkdir /var/www/example

mkdir /var/www/yoursite

mkdir /var/www/example/web

mkdir /var/www/example/ssl

mkdir /var/www/yoursite/web
```

```
mkdir /var/www/yoursite/ssl

echo "Include /etc/apache2/vhosts.conf" >> /etc/apache2/apache2.conf

vi /etc/apache2/vhosts.conf
```

The configuration file should look like this :

```
#NameVirtualHost 192.168.1.106:80
#<VirtualHost 192.168.1.106:80>
#       ServerName localhost
#       ServerAdmin root@localhost
#       DocumentRoot /var/www/sharedip
#</VirtualHost>
#NameVirtualHost 192.168.1.107:80
#<VirtualHost 192.168.1.107:80>
#       ServerName localhost
#       ServerAdmin root@localhost
#      DocumentRoot /var/www/sharedip
#</VirtualHost>

###EXAMPLE.COM###
<VirtualHost 192.168.1.106:80>
       ServerName example.com
       ServerAlias www.example.com
       DocumentRoot /var/www/example/web
       SetEnvIf Request_URI \.jpg dontlog
       SetEnvIf Request_URI \.gif dontlog
       SetEnvIf Request_URI \.css dontlog
       SetEnvIf Request_URI \.png dontlog
       SetEnvIf Request_URI \.txt dontlog
       CustomLog /var/log/apache2/example.com-access.log combined env=!dontlog
```

```
    ErrorLog /var/log/apache2/example.com-error.log
    Redirect /webmail http://www.example.com:81/squirrelmail
  Redirect /squirrelmail http://www.example.com:81/squirrelmail
    ErrorDocument 400 /error/invalidSyntax.html
    ErrorDocument 401 /error/authorizationRequired.html
    ErrorDocument 403 /error/forbidden.html
    ErrorDocument 404 /error/fileNotFound.html
    ErrorDocument 405 /error/methodNotAllowed.html
    ErrorDocument 500 /error/internalServerError.html
    ErrorDocument 503 /error/overloaded.html
</VirtualHost>
<VirtualHost 192.168.1.106:81>
    ServerName example.com
    ServerAlias www.example.com
    DocumentRoot /var/www/example/web
    SetEnvIf Request_URI \.jpg dontlog
    SetEnvIf Request_URI \.gif dontlog
    SetEnvIf Request_URI \.css dontlog
    SetEnvIf Request_URI \.png dontlog
    SetEnvIf Request_URI \.txt dontlog
    CustomLog /var/log/apache2/example.com-access.log combined env=!dontlog
    ErrorLog /var/log/apache2/example.com-error.log
    ErrorDocument 400 /error/invalidSyntax.html
    ErrorDocument 401 /error/authorizationRequired.html
    ErrorDocument 403 /error/forbidden.html
    ErrorDocument 404 /error/fileNotFound.html
    ErrorDocument 405 /error/methodNotAllowed.html
    ErrorDocument 500 /error/internalServerError.html
    ErrorDocument 503 /error/overloaded.html
</VirtualHost>
### Un-comment this part if you will use SSL on this virtual host.
### Dont forget to create or copy certificate files before you restart apache.
#<IfModule mod_ssl.c>
```

```
#<VirtualHost 192.168.1.106:443>
#       ServerName example.com
#       ServerAlias www.example.com
#       DocumentRoot /var/www/example/web
#       SetEnvIf Request_URI \.jpg dontlog
#       SetEnvIf Request_URI \.gif dontlog
#       SetEnvIf Request_URI \.css dontlog
#       SetEnvIf Request_URI \.png dontlog
#       SetEnvIf Request_URI \.txt dontlog
#       CustomLog /var/log/apache2/example.com-access.log combined env=!dontlog
#       ErrorLog /var/log/apache2/example.com-error.log
#       SSLEngine on
#       SSLCertificateFile /var/www/example/ssl/www.example.com.crt
#       SSLCertificateKeyFile /var/www/example/ssl/www.example.com.key
#       SSLCertificateChainFile /var/www/example/ssl/www.example.com.key.org
#       Redirect /webmail http://www.example.com:81/squirrelmail
#     Redirect /squirrelmail http://www.example.com:81/squirrelmail
#       ErrorDocument 400 /error/invalidSyntax.html
#       ErrorDocument 401 /error/authorizationRequired.html
#       ErrorDocument 403 /error/forbidden.html
#       ErrorDocument 404 /error/fileNotFound.html
#       ErrorDocument 405 /error/methodNotAllowed.html
#       ErrorDocument 500 /error/internalServerError.html
#       ErrorDocument 503 /error/overloaded.html
#</VirtualHost>
#</IfModule>
<VirtualHost 192.168.1.106:10001>
        ServerName example.com
        ServerAlias www.example.com
        DocumentRoot /var/www/example/web
        SetEnvIf Request_URI \.jpg dontlog
        SetEnvIf Request_URI \.gif dontlog
        SetEnvIf Request_URI \.css dontlog
```

```
    SetEnvIf Request_URI \.png dontlog
    SetEnvIf Request_URI \.txt dontlog
    CustomLog /var/log/apache2/example.com-access.log combined env=!dontlog
    ErrorLog /var/log/apache2/example.com-error.log
    Redirect /webmail http://www.example.com:81/squirrelmail
   Redirect /squirrelmail http://www.example.com:81/squirrelmail
    ErrorDocument 400 /error/invalidSyntax.html
    ErrorDocument 401 /error/authorizationRequired.html
    ErrorDocument 403 /error/forbidden.html
    ErrorDocument 404 /error/fileNotFound.html
    ErrorDocument 405 /error/methodNotAllowed.html
    ErrorDocument 500 /error/internalServerError.html
    ErrorDocument 503 /error/overloaded.html
</VirtualHost>
<VirtualHost 192.168.1.106:20001>
    ServerName example.com
    ServerAlias www.example.com
    DocumentRoot /var/www/example/web
    SetEnvIf Request_URI \.jpg dontlog
    SetEnvIf Request_URI \.gif dontlog
    SetEnvIf Request_URI \.css dontlog
    SetEnvIf Request_URI \.png dontlog
    SetEnvIf Request_URI \.txt dontlog
    CustomLog /var/log/apache2/example.com-access.log combined env=!dontlog
    ErrorLog /var/log/apache2/example.com-error.log
    Redirect /webmail http://www.example.com:81/squirrelmail
   Redirect /squirrelmail http://www.example.com:81/squirrelmail
    ErrorDocument 400 /error/invalidSyntax.html
    ErrorDocument 401 /error/authorizationRequired.html
    ErrorDocument 403 /error/forbidden.html
    ErrorDocument 404 /error/fileNotFound.html
    ErrorDocument 405 /error/methodNotAllowed.html
    ErrorDocument 500 /error/internalServerError.html
```

```
    ErrorDocument 503 /error/overloaded.html
</VirtualHost>
###YOURSITE.COM###
<VirtualHost 192.168.1.107:80>
    ServerName yoursite.com
    ServerAlias www.yoursite.com
    DocumentRoot /var/www/yoursite/web
    SetEnvIf Request_URI \.jpg dontlog
    SetEnvIf Request_URI \.gif dontlog
    SetEnvIf Request_URI \.css dontlog
    SetEnvIf Request_URI \.png dontlog
    SetEnvIf Request_URI \.txt dontlog
    CustomLog /var/log/apache2/yoursite.com-access.log combined env=!dontlog
    ErrorLog /var/log/apache2/yoursite.com-error.log
    Redirect /webmail http://www.yoursite.com:81/squirrelmail
  Redirect /squirrelmail http://www.yoursite.com:81/squirrelmail
    ErrorDocument 400 /error/invalidSyntax.html
    ErrorDocument 401 /error/authorizationRequired.html
    ErrorDocument 403 /error/forbidden.html
    ErrorDocument 404 /error/fileNotFound.html
    ErrorDocument 405 /error/methodNotAllowed.html
    ErrorDocument 500 /error/internalServerError.html
    ErrorDocument 503 /error/overloaded.html
</VirtualHost>
<VirtualHost 192.168.1.107:81>
    ServerName yoursite.com
    ServerAlias www.yoursite.com
    DocumentRoot /var/www/yoursite/web
    SetEnvIf Request_URI \.jpg dontlog
    SetEnvIf Request_URI \.gif dontlog
    SetEnvIf Request_URI \.css dontlog
    SetEnvIf Request_URI \.png dontlog
    SetEnvIf Request_URI \.txt dontlog
```

```
    CustomLog /var/log/apache2/yoursite.com-access.log combined env=!dontlog
    ErrorLog /var/log/apache2/yoursite.com-error.log
    ErrorDocument 400 /error/invalidSyntax.html
    ErrorDocument 401 /error/authorizationRequired.html
    ErrorDocument 403 /error/forbidden.html
    ErrorDocument 404 /error/fileNotFound.html
    ErrorDocument 405 /error/methodNotAllowed.html
    ErrorDocument 500 /error/internalServerError.html
    ErrorDocument 503 /error/overloaded.html
</VirtualHost>
### Un-comment this part if you will use SSL on this virtual host.
### Dont forget to create or copy certificate files before you restart apache.
#<IfModule mod_ssl.c>
#<VirtualHost 192.168.1.107:443>
#    ServerName yoursite.com
#    ServerAlias www.yoursite.com
#    DocumentRoot /var/www/yoursite/web
#    SetEnvIf Request_URI \.jpg dontlog
#    SetEnvIf Request_URI \.gif dontlog
#    SetEnvIf Request_URI \.css dontlog
#    SetEnvIf Request_URI \.png dontlog
#    SetEnvIf Request_URI \.txt dontlog
#    CustomLog /var/log/apache2/yoursite.com-access.log combined env=!dontlog
#    ErrorLog /var/log/apache2/yoursite.com-error.log
#    SSLEngine on
#    SSLCertificateFile /var/www/yoursite/ssl/www.yoursite.com.crt
#    SSLCertificateKeyFile /var/www/yoursite/ssl/www.yoursite.com.key
#    SSLCertificateChainFile /var/www/yoursite/ssl/www.yoursite.com.key.org
#    Redirect /webmail http://www.yoursite.com:81/squirrelmail
#   Redirect /squirrelmail http://www.yoursite.com:81/squirrelmail
#    ErrorDocument 400 /error/invalidSyntax.html
#    ErrorDocument 401 /error/authorizationRequired.html
#    ErrorDocument 403 /error/forbidden.html
```

```
#       ErrorDocument 404 /error/fileNotFound.html
#       ErrorDocument 405 /error/methodNotAllowed.html
#       ErrorDocument 500 /error/internalServerError.html
#       ErrorDocument 503 /error/overloaded.html
#</VirtualHost>
#</IfModule>
```

The "SetEnvIf Request_URI ... " and "env=!dontlog" options are there to make logs cleaner, they will not log access to images and css files.

I use ISPConfig shareip files and error pages : **HERE**

Modify them to your needs.

First install wget :

```
apt-get install wget
```

```
cd /var/www

wget http://www.blogama.org/errorpages.tar.gz

tar -zxvf errorpages.tar.gz

rm -rf errorpages.tar.gz

chmod 755 /var/www/error/ -R

cp -av /var/www/error/ /var/www/example/web/
```

```
cp -av /var/www/error/ /var/www/yoursite/web/
```

We will now edit the file /etc/apache2/sites-available/default so that it doesn't log ldirectord (the load balancer) requests :

```
mv /etc/apache2/sites-available/default /etc/apache2/sites-available/default.bak

vi /etc/apache2/sites-available/default
```

And paste this :

```
NameVirtualHost *
<VirtualHost *>
    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/
    <Directory />
        Options FollowSymLinks
        AllowOverride None
    </Directory>
    <Directory /var/www/>
        Options FollowSymLinks MultiViews
        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>
        ErrorLog /var/log/apache2/error.log
        # Possible values include: debug, info, notice, warn, error, crit,
        # alert, emerg.
        LogLevel warn
        #CustomLog /var/log/apache2/access.log combined
        SetEnvIf Request_URI "^/ldirectord\.php$" dontlog
        SetEnvIf Request_URI "token" dontlog
        SetEnvIf Remote_Addr "127\.0\.0\.1" dontlog
        SetEnvIf Request_URI \.ico dontlog
        CustomLog /var/log/apache2/access.log combined env=!dontlog
        ServerSignature On
    Alias /doc/ "/usr/share/doc/"
    <Directory "/usr/share/doc/">
        Options Indexes MultiViews FollowSymLinks
        AllowOverride None
        Order deny,allow
        Deny from all
        Allow from 127.0.0.0/255.0.0.0 ::1/128
    </Directory>
</VirtualHost>
```

Now we will open some ports in apache that will be used later on :

```
vi /etc/apache2/ports.conf
```

```
Listen 80
```

```
Listen 81
Listen 10001
Listen 20001
<IfModule mod_ssl.c>
   Listen 443
</IfModule>
```

Port 10001 and 20001 will be used for monitoring. Port 81 will be for webmail.

We must restart apache to apply the modifications we made :

```
/etc/init.d/apache2 restart
```

7.3 Preparing the apache nodes for load balancing

Finally we must configure our Apache cluster nodes web1.example.com and web2.example.com to accept requests on the virtual IP addresses 192.168.1.106 and 192.168.1.107.

```
apt-get install iproute
```

Add the following to `/etc/sysctl.conf`:

```
vi /etc/sysctl.conf
```

```
# Enable configuration of arp_ignore option
net.ipv4.conf.all.arp_ignore = 1
# When an arp request is received on eth0, only respond if that address is
# configured on eth0. In particular, do not respond if the address is
# configured on lo
net.ipv4.conf.eth0.arp_ignore = 1
# Ditto for eth1, add for all ARPing interfaces
#net.ipv4.conf.eth1.arp_ignore = 1

# Enable configuration of arp_announce option
net.ipv4.conf.all.arp_announce = 2
# When making an ARP request sent through eth0 Always use an address that
# is configured on eth0 as the source address of the ARP request. If this
# is not set, and packets are being sent out eth0 for an address that is on
# lo, and an arp request is required, then the address on lo will be used.
# As the source IP address of arp requests is entered into the ARP cache on
# the destination, it has the effect of announcing this address. This is
# not desirable in this case as adresses on lo on the real-servers should
# be announced only by the linux-director.
net.ipv4.conf.eth0.arp_announce = 2
# Ditto for eth1, add for all ARPing interfaces
#net.ipv4.conf.eth1.arp_announce = 2
```

Then run this:

```
sysctl -p
```

Add this section for the virtual IP address to `/etc/network/interfaces`:

```
vi /etc/network/interfaces
```

```
auto lo:0
iface lo:0 inet static
 address 192.168.1.106
 netmask 255.255.255.255
 pre-up sysctl -p > /dev/null
auto lo:1
iface lo:1 inet static
 address 192.168.1.107
 netmask 255.255.255.255
 pre-up sysctl -p > /dev/null
```

Then run this:

Please note: after the following step you will probably get this error: SIOCSIFFLAGS: Cannot assign requested address

That is a normal bug and you can ignore it.

```
ifup lo:0

ifup lo:1
```

If you change the IP at a later stage its recommended to do `ifup lo:0` then `ifdown lo:0` then again `ifup lo:0`

Finally we must create the file ldirectord.php. This file is requested by the two load balancer nodes repeatedly so that they can see if the two Apache nodes are still running along with MySQL. I assume that the document root of the main apache web site on web1 and web2 is `/var/www`, therefore we create the

file `/var/www/ldirectord.php`:

```
vi /var/www/ldirectord.php
```

and copy this :

```
<?php
$dbhost = 'localhost';
$dbuser = 'ldirectord';
$dbpass = 'LDIRECTORD_PASSWORD';
$conn = mysql_connect($dbhost, $dbuser, $dbpass) or die ('Error connecting to mysql');
print "Connected to MySQL";
?>
```

Later we will set up MySQL with local access to user "ldirectord", right now the file won't work.

8. DNS Server (web1, web2)8.1 Install the DNS Server

Run :

```
apt-get install bind9
```

For security reasons we want to run BIND chrooted so we have to do the following steps:

```
/etc/init.d/bind9 stop
```

Edit the file `/etc/default/bind9` so that the daemon will run as the unprivileged user bind, chrooted to `/var/lib/named`. Modify the line: `OPTIONS="-u bind"` so that it reads `OPTIONS="-u bind -t /var/lib/named"`:

```
vi /etc/default/bind9
```

```
OPTIONS="-u bind -t /var/lib/named"
# Set RESOLVCONF=no to not run resolvconf
RESOLVCONF=yes
```

Create the necessary directories under `/var/lib`:

```
mkdir -p /var/lib/named/etc

mkdir /var/lib/named/dev

mkdir -p /var/lib/named/var/cache/bind

mkdir -p /var/lib/named/var/run/bind/run
```

Then move the config directory from `/etc` to `/var/lib/named/etc`:

```
mv /etc/bind /var/lib/named/etc
```

Create a symlink to the new config directory from the old location (to avoid problems when bind gets updated in the future):

```
ln -s /var/lib/named/etc/bind /etc/bind
```

Make null and random devices, and fix permissions of the directories:

```
mknod /var/lib/named/dev/null c 1 3

mknod /var/lib/named/dev/random c 1 8

chmod 666 /var/lib/named/dev/null /var/lib/named/dev/random

chown -R bind:bind /var/lib/named/var/*

chown -R bind:bind /var/lib/named/etc/bind
```

We need to modify `/etc/default/syslogd` so that we can still get important messages logged to the system logs. Modify the line: `SYSLOGD=""` so that it reads: `SYSLOGD="-a /var/lib/named/dev/log"`:

```
vi /etc/default/syslogd
```

```
#
# Top configuration file for syslogd
#
#
```

```
# Full documentation of possible arguments are found in the manpage
# syslogd(8).
#
#
# For remote UDP logging use SYSLOGD="-r"
#
SYSLOGD="-a /var/lib/named/dev/log"
```

Restart the logging daemon:

```
/etc/init.d/sysklogd restart
```

Start up BIND, and check `/var/log/syslog` for errors:

```
/etc/init.d/bind9 start
```

8.2 Configure bind

We are going to configure bind with 2 domains, example.com which will be the nameserver and we will configure bind for yoursite.com as well.

Now the main configuration file in BIND is named.conf, however named.conf.local is already included in named.conf and its there for customized configuration, so we will edit named.conf.local and we will add our zones, here I added a zone camed tm.local as well as a reverse zone for 192.168.1.0:

```
vi /etc/bind/named.conf.local
```

```
#EXAMPLE.COM
zone "example.com" {
        type master;
        file "/etc/bind/zones/example.com.db";
        };
#YOURSITE.COM
zone "yoursite.com" {
        type master;
        file "/etc/bind/zones/yoursite.com.db";
        };
# This is the zone definition for reverse DNS. replace 1.168.192 with your network address in reverse notation - e.g my network address is 192.168.1.X
zone "1.168.192.in-addr.arpa." {
     type master;
     file "/etc/bind/zones/rev.1.168.192.in-addr.arpa";
};
```

Note : If your ISP is delegating you a subnet maps (let says ip 192.168.1.100 to 192.168.1.112) read this for the reverse zone (see Customer/User Zone File) :

http://www.zytrax.com/books/dns/ch9/reverse.html

8.3 Configure zones

```
mkdir /etc/bind/zones

vi /etc/bind/zones/example.com.db
```

and make it look like this :

```
$TTL	86400
@	IN	SOA	ns1.example.com. admin.example.com. (
                2008060902      ; serial, todays date + todays serial #
                28800           ; refresh, seconds
                7200            ; retry, seconds
                604800          ; expire, seconds
                86400 )         ; minimum, seconds
;
        NS      ns1.example.com.        ; Inet Address of name server 1
        NS      ns2.example.com.        ; Inet Address of name server 2
;
 MX     10 example.com.
example.com.    A       192.168.1.106
www             A       192.168.1.106
ns1             A       192.168.1.106
ns2             A       192.168.1.106
dom01           A       192.168.1.100
dom02           A       192.168.1.101
lb1             A       192.168.1.102
lb2             A       192.168.1.103
web1            A       192.168.1.104
web2            A       192.168.1.105

example.com.    TXT  "v=spf1 ip4:192.168.1.104 ip4:192.168.1.105 a ptr a:web1.example.com a:web2.example.com ~all"
```

Now we will create the zone for yoursite.com :

```
vi /etc/bind/zones/yoursite.com.db
```

Make it look like this :

```
$TTL	86400
@	IN	SOA	ns1.example.com. admin.yoursite.com. (
		2008060902	; serial, todays date + todays serial #
		28800		; refresh, seconds
		7200		; retry, seconds
		604800		; expire, seconds
		86400 )		; minimum, seconds
;
		NS	ns1.example.com.		; Inet Address of name server 1
		NS	ns2.example.com.		; Inet Address of name server 2
;
 MX	10 yoursite.com.
yoursite.com.	A	192.168.1.107
www		A	192.168.1.107

yoursite.com.	TXT	"v=spf1 ip4:192.168.1.104 ip4:192.168.1.105 a ptr a:web1.example.com a:web2.example.com ~all"
```

Now let's go ahead with the reverse zone.

```
vi /etc/bind/zones/rev.1.168.192.in-addr.arpa
```

```
$TTL	86400
@		IN	SOA	ns1.example.com. hostmaster.example.com. (
		2008060901	; serial, todays date + todays serial #
		28800	; Refresh
		7200	; Retry
		604800	; Expire
		86400)	; Minimum TTL
```

```
                NS      ns1.example.com.
                NS      ns2.example.com.
100     PTR     dom01.example.com.
101     PTR     dom02.example.com.
102     PTR     lb1.example.com.
103     PTR     lb2.example.com.
104     PTR     web1.example.com.
105     PTR     web2.example.com.
106     PTR     example.com.
107     PTR     yoursite.com.
```

Now configure the server to forward any requests to your ISP server so it case resolve external IPs.

```
vi /etc/bind/named.conf.options
```

Uncomment the forwarder section to look like this:

```
[...]
forwarders {
    # Replace the address below with the address of your ISP DNS server
    123.123.123.123;
};
[...]
```

8.4 Configure the server to use itself as DNS

```
vi /etc/resolv.conf
```

```
search example.com
nameserver localhost
```

We have to restart bind :

```
/etc/init.d/bind9 restart
```

8.5 Test the DNS server

We will first install dig which in included in the package dnsutils :

```
apt-get install dnsutils
```

Now we will see if our dns servers give us the right answers :

on web1

```
dig yoursite.com @192.168.1.105
```

on web2

```
dig yoursite.com @192.168.1.104
```

On both you should see an output like this :

```
;  DiG 9.4.2-P1  yoursite.com
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 4547
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 2, ADDITIONAL: 2
;; QUESTION SECTION:
;yoursite.com.                IN      A
;; ANSWER SECTION:
yoursite.com.         86400   IN      A       192.168.1.107
;; AUTHORITY SECTION:
yoursite.com.         15090   IN      NS      ns2.example.com.
yoursite.com.         15090   IN      NS      ns1.example.com.
;; ADDITIONAL SECTION:
ns2.example.com.      162439  IN      A       192.168.1.106
ns1.example.com.      162439  IN      A       192.168.1.106
;; Query time: 27 msec
;; WHEN: Sun Sep 21 19:07:17 2008
;; MSG SIZE  rcvd: 124
```

Now we will test reverse lookup :

on web1

```
dig -x 192.168.1.107 @192.168.1.105
```

on web2***

```
dig -x 192.168.1.107 @192.168.1.104
```

Output should be similar to this :

```
;  DiG 9.4.2-P1  -x 192.168.1.107
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 22614
;; flags: qr rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 2, ADDITIONAL: 2
;; QUESTION SECTION:
;107.1.168.192.in-addr.arpa.     IN      PTR
;; ANSWER SECTION:
;107.1.168.192.in-addr.arpa. 86400 IN PTR  yoursite.com.
;; AUTHORITY SECTION:
;1.168.192.in-addr.arpa. 86400 IN  NS      ns2.example.com.
;1.168.192.in-addr.arpa. 86400 IN  NS      ns1.example.com.
;; ADDITIONAL SECTION:
ns1.example.com.       162147  IN      A       192.168.1.106
ns2.example.com.       162147  IN      A       192.168.1.106
;; Query time: 88 msec
;; WHEN: Sun Sep 21 19:12:09 2008
;; MSG SIZE  rcvd: 172
```

More info how to use dig :

http://www.madboa.com/geek/dig/

9. Proftpd (web1, web2)9.1 Proftpd installation

In order to install Proftpd, run

```
apt-get install proftpd ucf
```

You will be asked a question:

Run proftpd: <-- standalone

9.2 Proftpd configuration

```
vi /etc/proftpd/proftpd.conf
```

For security reasons add the following lines to /etc/proftpd/proftpd.conf:

```
DefaultRoot ~
IdentLookups off
ServerIdent on "FTP Server ready."
```

Then restart Proftpd:

```
/etc/init.d/proftpd restart
```

10. MySQL replication (web1, web2)10.1 Mysql Installing MySQL 5.0

```
apt-get install mysql-server-5.0 mysql-client-5.0
```

To make sure that the replication can work, we must make MySQL listen on all interfaces, therefore we comment out the line `bind-address = 127.0.0.1` in `/etc/mysql/my.cnf` :

```
vi /etc/mysql/my.cnf
```

```
[...]
# Instead of skip-networking the default is now to listen only on
# localhost which is more compatible and is not less secure.
#bind-address          = 127.0.0.1
[...]
```

Restart MySQL afterwards :

```
/etc/init.d/mysql restart
```

On web1

Now we set up a replication user slave2_user that can be used by web2.example.com to access the MySQL database :

```
mysql -u root -p
```

On the MySQL shell, run the following commands :

```
GRANT REPLICATION SLAVE ON *.* TO 'slave2_user'@'%' IDENTIFIED BY 'slave2_password';

FLUSH PRIVILEGES;

quit;
```

On web2

Now we do the last two steps again on web2.example.com :

```
mysql -u root -p
```

```
GRANT REPLICATION SLAVE ON *.* TO 'slave1_user'@'%' IDENTIFIED BY 'slave1_password';

FLUSH PRIVILEGES;
```

```
quit;
```

On web1 AND web2

We will now create a database that will be used later for the mail server :

```
mysqladmin -u root -p create mail
```

Next, we go to the MySQL shell:

```
mysql -u root -p
```

On the MySQL shell, we create the user mail_admin with the passwort mail_admin_password (replace it with your own password) who has SELECT,INSERT,UPDATE,DELETE privileges on the mail database. This user will be used by Postfix and Courier to connect to the mail database:

```
GRANT SELECT, INSERT, UPDATE, DELETE ON mail.* TO 'mail_admin'@'localhost' IDENTIFIED BY 'mail_admin_password';

GRANT SELECT, INSERT, UPDATE, DELETE ON mail.* TO 'mail_admin'@'localhost.localdomain' IDENTIFIED BY 'mail_admin_password';

FLUSH PRIVILEGES;

quit;
```

10.2 Setting Up Replication

Now we set up master-master replication in `/etc/mysql/my.cnf`. The crucial configuration options for master-master replication are auto_increment_increment and auto_increment_offset:

- auto_increment_increment controls the increment between successive AUTO_INCREMENT values.
- auto_increment_offset determines the starting point for AUTO_INCREMENT column values.

Let's assume we have N MySQL nodes (N=2 in this example), then auto_increment_increment has the value N on all nodes, and each node must have a different value for auto_increment_offset (1, 2, ..., N).

Now let's configure our two MySQL nodes:

On web1

```
vi /etc/mysql/my.cnf
```

add the following lines right below "[mysqld]":

```
server-id = 1
replicate-same-server-id = 0
auto-increment-increment = 2
auto-increment-offset = 1
#use 192.168.1.105 if you didnt install a crossover cable on eth1
master-host = 192.168.0.105
master-user = slave1_user
master-password = slave1_password
master-connect-retry = 60
replicate-do-db = mail
log-bin = /var/log/mysql/mysql-bin.log
```

```
binlog-do-db = mail
relay-log = /var/lib/mysql/slave-relay.log
relay-log-index = /var/lib/mysql/slave-relay-log.index
```

and modify the line "max_binlog_size" :

```
[...]
max_binlog_size        = 500M
[...]
```

```
/etc/init.d/mysql restart
```

On web2

```
vi /etc/mysql/my.cnf
```

```
server-id = 2
replicate-same-server-id = 0
auto-increment-increment = 2
auto-increment-offset = 2
#use 192.168.1.104 if you didnt install a crossover cable on eth1
master-host = 192.168.0.104
master-user = slave2_user
master-password = slave2_password
master-connect-retry = 60
replicate-do-db = mail
```

```
log-bin = /var/log/mysql/mysql-bin.log
binlog-do-db = mail
relay-log = /var/lib/mysql/slave-relay.log
relay-log-index = /var/lib/mysql/slave-relay-log.index
```

and modify the line "max_binlog_size" :

```
[...]
max_binlog_size        = 500M
[...]
```

```
/etc/init.d/mysql restart
```

On web1 AND web2

Now we will start replication :

```
mysql -u root -p
```

On the MySQL shell, run the following commands:

```
reset master;

stop slave;
```

```
reset slave;
```

Now run the following :

On web1

```
CHANGE MASTER TO MASTER_HOST='192.168.0.105', MASTER_USER='slave1_user', MASTER_PASSWORD='slave1_password', MASTER_LOG_FILE='mysql-bin.000001',
MASTER_LOG_POS=98;
```

```
start slave;

quit;
```

On web2

```
CHANGE MASTER TO MASTER_HOST='192.168.0.104', MASTER_USER='slave2_user', MASTER_PASSWORD='slave2_password', MASTER_LOG_FILE='mysql-bin.000001',
MASTER_LOG_POS=98;
```

```
start slave;
```

```
quit;
```

Now replication should work.

10.3 Testing replication

On web1 AND web2

```
mysql -u root -p
```

On the MySQL shell, run the following commands:

```
show slave status \G;
```

There are 3 important lines in the output that should look like this :

```
[...]
Slave_IO_State: Waiting for master to send event
[...]
Slave_IO_Running: Yes
Slave_SQL_Running: Yes
Replicate_Do_DB: mail
[...]
```

Now you can quit mysql :

```
quit;
```

On web1

We will insert some data on web1.example.com for testing and that will serve in the next chapter for mail :

```
mysql -u root -p
```

On the MySQL shell, run the following commands:

```
use mail;
```

We will create the following tables :

```
CREATE TABLE domains (

domain varchar(50) NOT NULL,

transport varchar(128) NOT NULL default 'smtp:[192.168.1.104]',

PRIMARY KEY (domain) )
```

```
TYPE=MyISAM;
```

```
CREATE TABLE forwardings (
source varchar(80) NOT NULL,
destination TEXT NOT NULL,
PRIMARY KEY (source) )
TYPE=MyISAM;
```

```
CREATE TABLE users (
email varchar(80) NOT NULL,
password varchar(20) NOT NULL,
quota INT(10) DEFAULT '10485760',
PRIMARY KEY (email)
) TYPE=MyISAM;
```

```
CREATE TABLE transport (

domain varchar(128) NOT NULL default '',

transport varchar(128) NOT NULL default '',

UNIQUE KEY domain (domain)

) TYPE=MyISAM;
```

```
quit;
```

These freshly create tables should appear on web2 mail database as well, thanks to replication.

On web2

Now we will verify that :

```
mysql -u root -p
```

On the MySQL shell, run the following commands:

```
use mail;
```

```
show tables;
```

The output should be :

```
+----------------+
| Tables_in_mail |
+----------------+
| domains        |
| forwardings    |
| transport      |
| users          |
+----------------+
4 rows in set (0.00 sec)
```

```
quit;
```

If you see that replication is working.

10.4 Creating user for ldirectord

We will now create the user that will connect to the database in the `ldirectord.php` file.

On web1 AND web2

```
mysql -u root -p
```

On the MySQL shell, run the following commands:

```
GRANT USAGE ON * . * TO 'ldirectord'@'localhost' IDENTIFIED BY 'LDIRECTORD_PASSWORD';

quit;
```

Now when you go with your browser at addresses :

http://192.168.1.104/ldirectord.php

and

http://192.168.1.105/ldirectord.php

You should see :

Connected to MySQL

displayed on the screen.

11. Mail server (web1, web2)11.1 Install Postfix, Courier, Saslauthd, MySQL, phpMyAdmin

Install Postfix, Courier, Saslauthd, MySQL, phpMyAdmin :

```
apt-get install postfix postfix-mysql postfix-doc courier-authdaemon courier-authlib-mysql courier-pop courier-pop-ssl courier-imap
courier-imap-ssl libsasl2-2 libsasl2-modules libsasl2-modules-sql sasl2-bin libpam-mysql openssl phpmyadmin libpam-smbpass
```

You will be asked a few questions:

Create directories for web-based administration? <-- No
General type of mail configuration: <-- Internet Site
System mail name: <-- web1.example.com (or web2.example.com)
SSL certificate required <-- Ok
Web server to reconfigure automatically: <-- apache2

11.2 Configure Postfix

Now we have to tell Postfix where it can find all the information in the database. Therefore we have to create six text files. You will notice that I tell Postfix to connect to MySQL on the IP address 127.0.0.1 instead of localhost. This is because Postfix is running in a chroot jail and does not have access to the MySQL socket which it would try to connect if I told Postfix to use localhost. If I use 127.0.0.1 Postfix uses TCP networking to connect to MySQL which is no problem even in a chroot jail (the alternative would be to move the MySQL socket into the chroot jail which causes some other problems).

Now let's create our six text files.

```
vi /etc/postfix/mysql-virtual_domains.cf
```

```
user = mail_admin
password = mail_admin_password
dbname = mail
query = SELECT domain AS virtual FROM domains WHERE domain='%s'
hosts = 127.0.0.1
```

```
vi /etc/postfix/mysql-virtual_forwardings.cf
```

```
user = mail_admin
password = mail_admin_password
dbname = mail
query = SELECT destination FROM forwardings WHERE source='%s'
hosts = 127.0.0.1
```

```
vi /etc/postfix/mysql-virtual_mailboxes.cf
```

```
user = mail_admin
password = mail_admin_password
dbname = mail
query = SELECT CONCAT(SUBSTRING_INDEX(email,'@',-1),'/',SUBSTRING_INDEX(email,'@',1),'/') FROM users WHERE email='%s'
hosts = 127.0.0.1
```

```
vi /etc/postfix/mysql-virtual_email2email.cf
```

```
user = mail_admin
password = mail_admin_password
dbname = mail
query = SELECT email FROM users WHERE email='%s'
hosts = 127.0.0.1
```

```
vi /etc/postfix/mysql-virtual_transports.cf
```

```
user = mail_admin
password = mail_admin_password
dbname = mail
query = SELECT transport FROM transport WHERE domain='%s'
hosts = 127.0.0.1
```

```
vi /etc/postfix/mysql-virtual_transports_notactive.cf
```

```
user = mail_admin
password = mail_admin_password
dbname = mail
query = SELECT transport FROM domains WHERE domain='%s'
hosts = 127.0.0.1
```

Then change the permissions and the group of these files:

```
chmod o= /etc/postfix/mysql-virtual_*.cf

chgrp postfix /etc/postfix/mysql-virtual_*.cf
```

Now we create a user and group called vmail with the home directory /home/vmail. This is where all mail boxes will be stored.

```
groupadd -g 5000 vmail

useradd -g vmail -u 5000 vmail -d /home/vmail -m
```

```
passwd vmail
```

Next we do some Postfix configuration.

Make sure you put the right 'myhostname' and 'mydestination' on web2.example.com:

```
postconf -e 'myhostname = web1.example.com'

postconf -e 'mydestination = web1.example.com, localhost, localhost.localdomain'

postconf -e 'mynetworks = 127.0.0.0/8'

postconf -e 'virtual_alias_domains ='

postconf -e 'virtual_alias_maps = proxy:mysql:/etc/postfix/mysql-virtual_forwardings.cf, mysql:/etc/postfix/mysql-virtual_email2email.cf'

postconf -e 'virtual_mailbox_domains = proxy:mysql:/etc/postfix/mysql-virtual_domains.cf'

postconf -e 'virtual_mailbox_maps = proxy:mysql:/etc/postfix/mysql-virtual_mailboxes.cf'

postconf -e 'virtual_mailbox_base = /home/vmail'

postconf -e 'virtual_uid_maps = static:5000'

postconf -e 'virtual_gid_maps = static:5000'

postconf -e 'smtpd_sasl_auth_enable = yes'

postconf -e 'broken_sasl_auth_clients = yes'
```

```
postconf -e 'smtpd_sasl_authenticated_header = yes'

postconf -e 'smtpd_recipient_restrictions = permit_mynetworks, permit_sasl_authenticated, reject_unauth_destination'

postconf -e 'smtpd_use_tls = yes'

postconf -e 'smtpd_tls_cert_file = /etc/postfix/smtpd.cert'

postconf -e 'smtpd_tls_key_file = /etc/postfix/smtpd.key'

postconf -e 'transport_maps = proxy:mysql:/etc/postfix/mysql-virtual_transports.cf'

postconf -e 'virtual_create_maildirsize = yes'

postconf -e 'virtual_mailbox_extended = yes'

postconf -e 'proxy_read_maps = $local_recipient_maps $mydestination $virtual_alias_maps $virtual_alias_domains $virtual_mailbox_maps $virtual_mailbox_domains $relay_recipient_maps $relay_domains $canonical_maps $sender_canonical_maps $recipient_canonical_maps $relocated_maps $transport_maps $mynetworks'
```

Afterwards we create the SSL certificate that is needed for TLS:

```
cd /etc/postfix

openssl req -new -outform PEM -out smtpd.cert -newkey rsa:2048 -nodes -keyout smtpd.key -keyform PEM -days 365 -x509
```

Country Name (2 letter code) [AU]: <-- Enter your Country Name (e.g., "DE").
State or Province Name (full name) [Some-State]: <-- Enter your State or Province Name.

Locality Name (eg, city) []: <-- Enter your City.b
Organization Name (eg, company) [Internet Widgits Pty Ltd]: <-- Enter your Organization Name (e.g., the name of your company).
Organizational Unit Name (eg, section) []: <-- Enter your Organizational Unit Name (e.g. "IT Department").
Common Name (eg, YOUR name) []: <-- Enter the Fully Qualified Domain Name of the system (e.g. "server1.example.com").
Email Address []: <-- Enter your Email Address.

Then change the permissions of the smtpd.key:

```
chmod o= /etc/postfix/smtpd.key
```

11.3 Solution to local mail problem on web2.example.com

When we will configure ldirectord, only web1.example.com will be active for SMTP (port 25). Our second server will be on standby and will take the active role only if postfix fails on web1.example.com.

The reason why we do that is because if both servers are active on port 25, half of mail from the outside will go on web1 and the other half on web2. It would be a nightmare to sync...

This works fine for mail from the outside but for local mail on web2.example.com will be delivered locally. In other words, let say you have a "contact us" form on your website that send an email and the visitor is on web2.example.com, the mail will never reach web1.example.com where all the mail should go. This happend because when a mail is sent from web2.example.com to sales@example.com for example, it ask the DNS server what is the mail server address of example.com, get an answer of 192.168.1.106 which is himself so the mail never leave the server.

The trick is to use postfix transport to send local mail to web1.example.com.

To achieve this, we will use a bash script that will run every minute which will send local mail to web1.example.com :

web2.example.com

```
vi /root/check_smtp
```

```
#!/bin/bash
# Local mail fix for load balancing
# Copyright (c) 2008 blogama.org
# This script is licensed under GNU GPL version 2.0 or above
# -------------------------------------------------------------------------
### The purpose of this script is to fix local mail problem with load balancing ###
### If someone can make this script work with postconf -e instead of sed (ugly)
### to modify /etc/postfix/main.cf WITH CRONTAB let me know, did not work for me
### To be modified ###
MASTERSERVER="web1.example.com"
###### Do not make modifications below ######
### Binaries ###
MAIL=$(which mail)
TELNET=$(which telnet)
#This
POSTCONF="/etc/postfix/main.cf"
### Check if server 1 is responding on smtp ###
(
echo "quit"
) | $TELNET $MASTERSERVER 25 | grep Connected > /dev/null 2>&1
if [ "$?" -ne "1" ]; then
  ### If in a previous attempt web1 was not connecting but now connect, web2 will forward all local mail to web1 ###
  if [ -f smtpactive ]; then
    sed -i 's/transport_maps = .*/transport_maps = proxy:mysql:\/etc\/postfix\/mysql-virtual_transports_notactive.cf/' $POSTCONF
    /etc/init.d/postfix restart
    rm /root/smtpactive
  ### If in a previous attempt web1 was connecting and still does, do nothing and exit ###
  else
    exit 1;
  fi
```

```
else
  ### If in a previous attempt web1 was not connecting and still doesnt, web2 is already active for local mail, do nothing and exit ###
  cd /root
  if [ -f smtpactive ]; then
    exit 1;
  fi
  ### If in a previous attempt web1 was connecting but now doesnt, web2 will take active role for local mail ###
  echo "SMTP active on web2" > /root/smtpactive
  sed -i 's/transport_maps = .*/transport_maps = proxy:mysql:\/etc\/postfix\/mysql-virtual_transports.cf/' $POSTCONF
  /etc/init.d/postfix restart
fi
```

```
chmod +x /root/check_smtp
```

Now we will forward all local mail to web1.example.com by doing the following :

```
postconf -e 'transport_maps = proxy:mysql:/etc/postfix/mysql-virtual_transports_notactive.cf

/etc/init.d/postfix restart
```

and add the script to your crontab :

```
crontab -e
```

```
[...]
* * * * * /root/check_smtp >/dev/null 2>&1
[...]
```

11.4 Configure Saslauthd

```
mkdir -p /var/spool/postfix/var/run/saslauthd
```

Then edit `/etc/default/saslauthd`. Set `START` to `yes` and change the line `OPTIONS="-c -m /var/run/saslauthd"` to `OPTIONS="-c -m /var/spool/postfix/var/run/saslauthd -r"`:

```
vi /etc/default/saslauthd
```

```
#
# Settings for saslauthd daemon
# Please read /usr/share/doc/sasl2-bin/README.Debian for details.
#
# Should saslauthd run automatically on startup? (default: no)
START=yes
# Description of this saslauthd instance. Recommended.
# (suggestion: SASL Authentication Daemon)
DESC="SASL Authentication Daemon"
# Short name of this saslauthd instance. Strongly recommended.
# (suggestion: saslauthd)
NAME="saslauthd"
```

```
# Which authentication mechanisms should saslauthd use? (default: pam)
#
# Available options in this Debian package:
# getpwent  -- use the getpwent() library function
# kerberos5 -- use Kerberos 5
# pam       -- use PAM
# rimap     -- use a remote IMAP server
# shadow    -- use the local shadow password file
# sasldb    -- use the local sasldb database file
# ldap      -- use LDAP (configuration is in /etc/saslauthd.conf)
#
# Only one option may be used at a time. See the saslauthd man page
# for more information.
#
# Example: MECHANISMS="pam"
MECHANISMS="pam"
# Additional options for this mechanism. (default: none)
# See the saslauthd man page for information about mech-specific options.
MECH_OPTIONS=""
# How many saslauthd processes should we run? (default: 5)
# A value of 0 will fork a new process for each connection.
THREADS=5
# Other options (default: -c -m /var/run/saslauthd)
# Note: You MUST specify the -m option or saslauthd won't run!
#
# See /usr/share/doc/sasl2-bin/README.Debian for Debian-specific information.
# See the saslauthd man page for general information about these options.
#
# Example for postfix users: "-c -m /var/spool/postfix/var/run/saslauthd"
#OPTIONS="-c -m /var/run/saslauthd"
OPTIONS="-c -m /var/spool/postfix/var/run/saslauthd -r"
```

Then create the file `/etc/pam.d/smtp`. It should contain only the following two lines (go sure to fill in your correct database details):

```
vi /etc/pam.d/smtp
```

```
auth    required   pam_mysql.so user=mail_admin passwd=mail_admin_password host=127.0.0.1 db=mail table=users usercolumn=email passwdcolumn=password crypt=1
account sufficient pam_mysql.so user=mail_admin passwd=mail_admin_password host=127.0.0.1 db=mail table=users usercolumn=email passwdcolumn=password crypt=1
```

Next create the file `/etc/postfix/sasl/smtpd.conf`. It should look like this:

```
vi /etc/postfix/sasl/smtpd.conf
```

```
pwcheck_method: saslauthd
mech_list: plain login
allow_plaintext: true
auxprop_plugin: mysql
sql_hostnames: 127.0.0.1
sql_user: mail_admin
sql_passwd: mail_admin_password
sql_database: mail
sql_select: select password from users where email = '%u'
```

Next add the `postfix` user to the `sasl` group (this makes sure that Postfix has the permission to access saslauthd):

```
adduser postfix sasl
```

Then restart Postfix and Saslauthd:

```
/etc/init.d/postfix restart

/etc/init.d/saslauthd restart
```

11.5 Configure Courier

Now we have to tell Courier that it should authenticate against our MySQL database. First, edit `/etc/courier/authdaemonrc` and change the value of `authmodulelist` so that it reads:

```
vi /etc/courier/authdaemonrc
```

```
[...]
authmodulelist="authmysql"
[...]
```

Then make a backup of `/etc/courier/authmysqlrc` and empty the old file:

```
cp /etc/courier/authmysqlrc /etc/courier/authmysqlrc_orig

cat /dev/null > /etc/courier/authmysqlrc
```

Then open `/etc/courier/authmysqlrc` and put the following lines into it:

```
vi /etc/courier/authmysqlrc
```

```
MYSQL_SERVER localhost
MYSQL_USERNAME mail_admin
MYSQL_PASSWORD mail_admin_password
MYSQL_PORT 0
MYSQL_DATABASE mail
MYSQL_USER_TABLE users
MYSQL_CRYPT_PWFIELD password
#MYSQL_CLEAR_PWFIELD password
MYSQL_UID_FIELD 5000
MYSQL_GID_FIELD 5000
MYSQL_LOGIN_FIELD email
MYSQL_HOME_FIELD "/home/vmail"
MYSQL_MAILDIR_FIELD CONCAT(SUBSTRING_INDEX(email,'@',-1),'/',SUBSTRING_INDEX(email,'@',1),'/')
#MYSQL_NAME_FIELD
#MYSQL_QUOTA_FIELD quota
```

Then restart Courier:

```
/etc/init.d/courier-authdaemon restart

/etc/init.d/courier-imap restart

/etc/init.d/courier-imap-ssl restart

/etc/init.d/courier-pop restart

/etc/init.d/courier-pop-ssl restart
```

By running

```
telnet localhost pop3
```

you can see if your POP3 server is working correctly. It should give back `+OK Hello there`. (Type quit to get back to the Linux shell.)

```
root@server1:/etc/postfix# telnet localhost pop3
Trying 127.0.0.1...
Connected to localhost.localdomain.
Escape character is '^]'.
+OK Hello there.
quit
+OK Better luck next time.
Connection closed by foreign host.
```

11.6 Modify /etc/aliases

Now we should open `/etc/aliases`. Make sure that postmaster points to root and root to your own username or your email address, e.g. like this :

```
vi /etc/aliases
```

```
[...]
postmaster: root
root: postmaster@example.com
[...]
```

Whenever you modify `/etc/aliases`, you must run

```
newaliases
```

afterwards and restart Postfix:

```
/etc/init.d/postfix restart
```

11.7 Spamassassin

Now we will install Spamassassin:

```
apt-get install spamassassin spamc
```

We want it to run as non-root, so add a spamd user and group:

```
groupadd spamd

useradd -g spamd -s /bin/false -d /var/log/spamassassin spamd

mkdir /var/log/spamassassin

chown spamd:spamd /var/log/spamassassin
```

Edit `/etc/default/spamassassin` so these options are set :

```
vi /etc/default/spamassassin
```

```
# /etc/default/spamassassin
# Duncan Findlay
# WARNING: please read README.spamd before using.
# There may be security risks.
# Change to one to enable spamd
ENABLED=1
# Options
# See man spamd for possible options. The -d option is automatically added.
# SpamAssassin uses a preforking model, so be careful! You need to
# make sure --max-children is not set to anything higher than 5,
# unless you know what you're doing.
#OPTIONS="--create-prefs --max-children 5 --helper-home-dir"
SAHOME="/var/log/spamassassin/"
OPTIONS="--create-prefs --max-children 5 --username spamd -H ${SAHOME} -s ${SAHOME}spamd.log"
# Pid file
# Where should spamd write its PID to file? If you use the -u or
# --username option above, this needs to be writable by that user.
# Otherwise, the init script will not be able to shut spamd down.
PIDFILE="/var/run/spamd.pid"
# Set nice level of spamd
#NICE="--nicelevel 15"
# Cronjob
# Set to anything but 0 to enable the cron job to automatically update
# spamassassin's rules on a nightly basis
```

```
CRON=0
```

Start the spamassassin daemon:

```
/etc/init.d/spamassassin start
```

We will add spamassassin to postfix by doing the following :

```
vi /etc/postfix/master.cf
```

Edit the first line of the configuration file so that it looks like this :

```
smtp      inet  n       -       -       -       -       smtpd -o content_filter=spamassassin
```

And add this to the end of the file :

```
spamassassin unix -     n       n       -       -       pipe
        user=spamd argv=/usr/bin/spamc -f -e
        /usr/sbin/sendmail -oi -f ${sender} ${recipient}
```

11.8 Test Postfix

To see if Postfix is ready for SMTP-AUTH and TLS, run

```
telnet localhost 25
```

After you have established the connection to your Postfix mail server type

```
ehlo localhost
```

If you see the lines

`250-STARTTLS` and `250-AUTH LOGIN PLAIN`

everything is fine.

Now type

```
quit
```

to return to the shell.

11.9 Populate the database

On either server do (not both!) :

```
mysql -u root -p
```

```
USE mail;
INSERT INTO `domains` (`domain`) VALUES ('example.com');

INSERT INTO `users` (`email`, `password`) VALUES ('sales@example.com', ENCRYPT('secret'));

quit;
```

NEVER FORGET TO USE MYSQL ENCRYPT FUNCTION FOR PASSWORD!

11.10 Send A Welcome Email For Creating Maildir

When you create a new email account and try to fetch emails from it (with POP3/IMAP) you will probably get error messages saying that the Maildir doesn't exist. The Maildir is created automatically when the first email arrives for the new account. Therefore it's a good idea to send a welcome email to a new account.

First, we install the mailx package:

```
apt-get install mailx
```

To send a welcome email to sales@example.com, we do this:

```
mailx sales@example.com
```

You will be prompted for the subject. Type in the subject (e.g. Welcome), then press ENTER, and in the next line type your message. When the message is finished, press ENTER again so that you are in a new line, then press CTRL+D; if you don't want to cc the mail, press ENTER again:

```
root@server1:/usr/local/sbin# mailx sales@example.comSubject: Welcome <-- ENTERWelcome! Have fun with your new mail account.
<-- ENTER<-- CTRL+DCc: <-- ENTER
```

11.11 Installing SquirrelMail

SquirrelMail is a webmail interface that will let your users send and receive emails in a browser. This chapter shows how to install it and adjust it to our setup so that users can even change their email account password from the SquirrelMail interface.

To install SquirrelMail, we run:

```
apt-get install squirrelmail php-pear
```

Next we copy the Apache configuration that comes with the SquirrelMail package to the `/etc/apache2/conf.d` directory and restart Apache:

```
cp /etc/squirrelmail/apache.conf /etc/apache2/conf.d/squirrelmail.conf

/etc/init.d/apache2 restart
```

SquirrelMail comes with some pre-installed plugins, unfortunately none of them is capable of letting us change our email password in our MySQL database. But there's the Change SQL Password plugin which we can install manually:

The plugin depends on the Pear-DB package so we install it:

```
pear install DB
```

Then we install the Change SQL Password plugin itself:

```
cd /usr/share/squirrelmail/plugins

wget http://www.squirrelmail.org/countdl.php?fileurl=http%3A%2F%2Fwww.squirrelmail.org%2Fplugins%2Fchange_sqlpass-3.3-1.2.tar.gz

tar xvfz change_sqlpass-3.3-1.2.tar.gz

cd change_sqlpass

cp config.php.sample config.php
```

Now we must edit config.php and adjust it to our setup. Please adjust the $csp_dsn, $lookup_password_query, $password_update_queries, $password_encryption, $csp_salt_static, and $csp_delimiter variables as follows and comment out $csp_salt_query:

```
mv /usr/share/squirrelmail/plugins/change_sqlpass/config.php /usr/share/squirrelmail/plugins/change_sqlpass/config.php.bak

vi /usr/share/squirrelmail/plugins/change_sqlpass/config.php
```

and copy paste this :

```
<?php
/**
```

```
* SquirrelMail Change SQL Password Plugin
* Copyright (C) 2001-2002 Tyler Akins
*             2002 Thijs Kinkhorst
*             2002-2005 Paul Lesneiwski
* This program is licensed under GPL. See COPYING for details
*
* @package plugins
* @subpackage Change SQL Password
*
*/

// Global Variables, don't touch these unless you want to break the plugin
//
global $csp_dsn, $password_update_queries, $lookup_password_query,
      $force_change_password_check_query, $password_encryption,
      $csp_salt_query, $csp_salt_static, $csp_secure_port,
      $csp_non_standard_http_port, $csp_delimiter, $csp_debug,
      $min_password_length, $max_password_length, $include_digit_in_password,
      $include_uppercase_letter_in_password, $include_lowercase_letter_in_password,
      $include_nonalphanumeric_in_password;
$csp_dsn = 'mysql://mail_admin:mail_admin_password@localhost/mail';
$lookup_password_query = 'SELECT count(*) FROM users WHERE email = "%1" AND password = %4';
$password_update_queries = array('UPDATE users SET password = %4 WHERE email = "%1"');
$force_change_password_check_query = '';
$password_encryption = 'MYSQLENCRYPT';
$csp_salt_static = 'LEFT(password, 2)';
$csp_secure_port = 0;
$csp_non_standard_http_port = 0;
$min_password_length = 6;
$max_password_length = 0;
$include_digit_in_password = 0;
$include_uppercase_letter_in_password = 0;
$include_lowercase_letter_in_password = 0;
```

```
  $include_nonalphanumeric_in_password = 0;
  $csp_delimiter = '@';
  $csp_debug = 0;
?>
```

For reference, the lines that have been changed are the following :

```
[...]
$csp_dsn = 'mysql://mail_admin:mail_admin_password@localhost/mail';
[...]
$lookup_password_query = 'SELECT count(*) FROM users WHERE email = "%1" AND password = %4';
[...]
$password_update_queries = array('UPDATE users SET password = %4 WHERE email = "%1"');
[...]
$password_encryption = 'MYSQLENCRYPT';
[...]
$csp_salt_static = 'LEFT(password, 2)';
[...]
//$csp_salt_query = 'SELECT salt FROM users WHERE username = "%1"';
[...]
$csp_delimiter = '@';
[...]
```

The Change SQL Password plugin also depends on the Compatibility plugin which we install as follows:

```
cd /usr/share/squirrelmail/plugins

wget http://www.squirrelmail.org/countdl.php?fileurl=http%3A%2F%2Fwww.squirrelmail.org%2Fplugins%2Fcompatibility-2.0.11-1.0.tar.gz

tar xvfz compatibility-2.0.11-1.0.tar.gz
```

Now we must go into the SquirrelMail configuration and tell SquirrelMail that we use Courier as our POP3 and IMAP server and enable the Change SQL Password and the Compatibility plugins:

```
/usr/sbin/squirrelmail-configure
```

You'll see the following menu. Navigate through it as indicated:

```
SquirrelMail Configuration : Read: config.php (1.4.0)
---------------------------------------------------------
Main Menu --
1.  Organization Preferences
2.  Server Settings
3.  Folder Defaults
4.  General Options
5.  Themes
6.  Address Books
7.  Message of the Day (MOTD)
8.  Plugins
9.  Database
10. Languages
D.  Set pre-defined settings for specific IMAP servers
C   Turn color on
S   Save data
Q   Quit
Command >> <-- D
```

```
SquirrelMail Configuration : Read: config.php
---------------------------------------------------------
While we have been building SquirrelMail, we have discovered some
preferences that work better with some servers that don't work so
well with others.  If you select your IMAP server, this option will
set some pre-defined settings for that server.
Please note that you will still need to go through and make sure
everything is correct.  This does not change everything.  There are
only a few settings that this will change.
Please select your IMAP server:
   bincimap    = Binc IMAP server
   courier     = Courier IMAP server
   cyrus       = Cyrus IMAP server
   dovecot     = Dovecot Secure IMAP server
   exchange    = Microsoft Exchange IMAP server
   hmailserver = hMailServer
   macosx      = Mac OS X Mailserver
   mercury32   = Mercury/32
   uw          = University of Washington's IMAP server
   quit        = Do not change anything
Command >> <-- courier
```

```
 imap_server_type = courier
       default_folder_prefix = INBOX.
              trash_folder = Trash
               sent_folder = Sent
              draft_folder = Drafts
```

```
          show_prefix_option = false
        default_sub_of_inbox = false
show_contain_subfolders_option = false
          optional_delimiter = .
               delete_folder = true
Press any key to continue... <-- press some key
```

```
SquirrelMail Configuration : Read: config.php (1.4.0)
---------------------------------------------------------
Main Menu --
1.  Organization Preferences
2.  Server Settings
3.  Folder Defaults
4.  General Options
5.  Themes
6.  Address Books
7.  Message of the Day (MOTD)
8.  Plugins
9.  Database
10. Languages
D.  Set pre-defined settings for specific IMAP servers
C   Turn color on
S   Save data
Q   Quit
Command >> <-- 8
```

```
SquirrelMail Configuration : Read: config.php (1.4.0)
---------------------------------------------------------
Plugins
  Installed Plugins
  Available Plugins:
    1. abook_take
    2. administrator
    3. bug_report
    4. calendar
    5. change_sqlpass
    6. compatibility
    7. delete_move_next
    8. demo
    9. filters
    10. fortune
    11. info
    12. listcommands
    13. mail_fetch
    14. message_details
    15. newmail
    16. sent_subfolders
    17. spamcop
    18. squirrelspell
    19. test
    20. translate
R   Return to Main Menu
C   Turn color on
S   Save data
Q   Quit
Command >> <-- 6 (or whatever number the compatibility plugin has - it's needed by the change_sqlpass plugin)
```

```
SquirrelMail Configuration : Read: config.php (1.4.0)
---------------------------------------------------------
Plugins
  Installed Plugins
    1. compatibility
  Available Plugins:
    2. abook_take
    3. administrator
    4. bug_report
    5. calendar
    6. change_sqlpass
    7. delete_move_next
    8. demo
    9. filters
    10. fortune
    11. info
    12. listcommands
    13. mail_fetch
    14. message_details
    15. newmail
    16. sent_subfolders
    17. spamcop
    18. squirrelspell
    19. test
    20. translate
R   Return to Main Menu
C   Turn color on
S   Save data
Q   Quit
Command >> <-- 6 (the number of the change_sqlpass plugin)
```

```
SquirrelMail Configuration : Read: config.php (1.4.0)
---------------------------------------------------------
Plugins
  Installed Plugins
    1. compatibility
    2. change_sqlpass
  Available Plugins:
    3. abook_take
    4. administrator
    5. bug_report
    6. calendar
    7. delete_move_next
    8. demo
    9. filters
    10. fortune
    11. info
    12. listcommands
    13. mail_fetch
    14. message_details
    15. newmail
    16. sent_subfolders
    17. spamcop
    18. squirrelspell
    19. test
    20. translate
R   Return to Main Menu
C   Turn color on
S   Save data
Q   Quit
Command >> <-- S
```

```
SquirrelMail Configuration : Read: config.php (1.4.0)
---------------------------------------------------------
Plugins
  Installed Plugins
    1. compatibility
    2. change_sqlpass
  Available Plugins:
    3. abook_take
    4. administrator
    5. bug_report
    6. calendar
    7. delete_move_next
    8. demo
    9. filters
    10. fortune
    11. info
    12. listcommands
    13. mail_fetch
    14. message_details
    15. newmail
    16. sent_subfolders
    17. spamcop
    18. squirrelspell
    19. test
    20. translate
R   Return to Main Menu
C   Turn color on
S   Save data
Q   Quit
Command >> S
Data saved in config.php
```

```
Press enter to continue... <-- press some key
```

```
SquirrelMail Configuration : Read: config.php (1.4.0)
---------------------------------------------------------
Plugins
  Installed Plugins
    1. compatibility
    2. change_sqlpass
  Available Plugins:
    3. abook_take
    4. administrator
    5. bug_report
    6. calendar
    7. delete_move_next
    8. demo
    9. filters
    10. fortune
    11. info
    12. listcommands
    13. mail_fetch
    14. message_details
    15. newmail
    16. sent_subfolders
    17. spamcop
    18. squirrelspell
    19. test
    20. translate
R   Return to Main Menu
C   Turn color on
```

```
S   Save data
Q   Quit
Command >> <-- Q
```

Now you can type in *http://192.168.1.104/squirrelmail* in your browser to access SquirrelMail.

Log in with your email address (e.g. sales@example.com) and your password

Password can be changed in *Options --> Change Password.*

12. Setting up the load balancers (lb1, lb2)

12.1 Enable IPVS On The Load Balancers

First we must enable IPVS on our load balancers. IPVS (IP Virtual Server) implements transport-layer load balancing inside the Linux kernel, so called Layer-4 switching.

```
echo ip_vs_dh >> /etc/modules

echo ip_vs_ftp >> /etc/modules

echo ip_vs >> /etc/modules

echo ip_vs_lblc >> /etc/modules

echo ip_vs_lblcr >> /etc/modules

echo ip_vs_lc >> /etc/modules

echo ip_vs_nq >> /etc/modules

echo ip_vs_rr >> /etc/modules
```

```
echo ip_vs_sed >> /etc/modules

echo ip_vs_sh >> /etc/modules

echo ip_vs_wlc >> /etc/modules

echo ip_vs_wrr >> /etc/modules
```

Then we do this:

```
modprobe ip_vs_dh

modprobe ip_vs_ftp

modprobe ip_vs

modprobe ip_vs_lblc

modprobe ip_vs_lblcr

modprobe ip_vs_lc

modprobe ip_vs_nq

modprobe ip_vs_rr

modprobe ip_vs_sed

modprobe ip_vs_sh
```

```
modprobe ip_vs_wlc

modprobe ip_vs_wrr
```

12.2 Install Ultra Monkey (packages) On The Load Balancers

Install Ultra Monkey (packages) on the load balancers by doing the following :

```
apt-get install ipvsadm ldirectord heartbeat
```

12.3 Enable Packet Forwarding On The Load Balancers

The load balancers must be able to route traffic to the Apache nodes. Therefore we must enable packet forwarding on the load balancers. Add the following lines to `/etc/sysctl.conf` :

```
vi /etc/sysctl.conf
```

```
# Enables packet forwarding
net.ipv4.ip_forward = 1
```

Then do this:

```
sysctl -p
```

12.4 Configure heartbeat And ldirectord

Now we have to create three configuration files for heartbeat (carefull with space and tabs if you edit in some text editors, ldirectord is very picky!) :

on lb1 and lb2

```
vi /etc/ha.d/ha.cf
```

```
logfacility local0
bcast eth0 # Linux
mcast eth0 225.0.0.1 694 1 0
auto_failback on
node lb1.example.com
node lb2.example.com
respawn hacluster /usr/lib/heartbeat/ipfail
apiauth ipfail gid=haclient uid=hacluster
```

Important: As nodenames we must use the output of both :

```
uname -n
```

```
vi /etc/ha.d/haresources
```

```
lb1.example.com \
 ldirectord::ldirectord.cf \
```

```
LVSSyncDaemonSwap::master \
IPaddr2::192.168.1.106/24/eth0/192.168.1.255 \
IPaddr2::192.168.1.107/24/eth0/192.168.1.255
```

IPs 192.168.1.106 and 107 will be used later for websites (example.com and yoursite.com).

This file should be the same on both nodes, no matter if you start to create the file on lb1 or lb2!

```
vi /etc/ha.d/authkeys
```

```
auth 3 3 md5 somerandomstring
```

somerandomstring is a password which the two heartbeat daemons on lb1 and lb2 use to authenticate against each other. Use your own string here. You have the choice between three authentication mechanisms. I use md5 as I believe it is the most secure one.

/etc/ha.d/authkeys should be readable by root only, therefore we do this:

```
chmod 600 /etc/ha.d/authkeys
```

ldirectord is the actual load balancer. We are going to configure our two load balancers (lb1.example.com and lb2.example.com) in an active/passive setup, which means we have one active load balancer, and the other one is a hot-standby and becomes active if the active one fails. To make it work, we must create the ldirectord configuration file */etc/ha.d/ldirectord.cf* which again must be identical on lb1 and lb2.

```
vi /etc/ha.d/ldirectord.cf
```

```
checktimeout=5
checkinterval=5
autoreload=no
logfile="/var/log/ldirectord.log"
quiescent=no
#fork=yes

#FOR SMTP
virtual=192.168.1.106:25
        real=192.168.1.104:25 gate
        fallback=192.168.1.105:25 gate
        service=none
        scheduler=wlc
        protocol=tcp
        checktype=connect
virtual=192.168.1.107:25
        real=192.168.1.104:25 gate
        fallback=192.168.1.105:25 gate
        service=none
        scheduler=wlc
        protocol=tcp
        checktype=connect

#FOR DNS - CONNECT DOESNT WORK, MUST BE PATCHED BUT PING IS OK
virtual=192.168.1.106:53
        real=192.168.1.104:53 gate
        fallback=192.168.1.105:53 gate
        service=none
        scheduler=wlc
        checktype=ping
        protocol=udp
virtual=192.168.1.106:53
        real=192.168.1.104:53 gate
```

```
        fallback=192.168.1.105:53 gate
        service=dns
        scheduler=wlc
        checktype=ping
        protocol=tcp
#FOR HTTP
virtual=192.168.1.106:80
        real=192.168.1.104:80 gate
        real=192.168.1.105:80 gate
        service=http
        request="ldirectord.php"
        receive="Connected to MySQL"
        scheduler=wlc
        protocol=tcp
        checktype=negotiate
        persistent=28800
virtual=192.168.1.107:80
        real=192.168.1.104:80 gate
        real=192.168.1.105:80 gate
        service=http
        request="ldirectord.php"
        receive="Connected to MySQL"
        scheduler=wlc
        protocol=tcp
        checktype=negotiate
        persistent=28800

#FOR  WEBMAIL
virtual=192.168.1.106:81
        real=192.168.1.104:81 gate
        fallback=192.168.1.105:81 gate
        service=http
        request="ldirectord.php"
```

```
        receive="Connected to MySQL"
        scheduler=wlc
        protocol=tcp
        checktype=negotiate
virtual=192.168.1.107:81
        real=192.168.1.104:81 gate
        fallback=192.168.1.105:81 gate
        service=http
        request="ldirectord.php"
        receive="Connected to MySQL"
        scheduler=wlc
        protocol=tcp
        checktype=negotiate
#FOR POP3
virtual=192.168.1.106:110
        real=192.168.1.104:110 gate
        fallback=192.168.1.105:110 gate
        service=pop
        checktype = connect
        scheduler=wlc
        protocol=tcp
#FOR IMAP
virtual=192.168.1.106:143
        real=192.168.1.104:143 gate
        fallback=192.168.1.105:143 gate
        service=imap
        scheduler=wlc
        protocol=tcp
#FOR HTTPS
###Un-comment this part if you will use HTTPS
#virtual=192.168.1.106:443
#       real=192.168.1.104:443 gate
#       real=192.168.1.105:443 gate 2
```

```
#       service=http
#       request="ldirectord.php"
#       receive="Connected to MySQL"
#       scheduler=wlc
#       protocol=tcp
#       checktype=negotiate
#       persistent=28800
#
#virtual=192.168.1.107:443
#       real=192.168.1.104:443 gate
#       real=192.168.1.105:443 gate 2
#       service=http
#       request="ldirector.html"
#       receive="Test Page"
#       scheduler=wlc
#       protocol=tcp
#       checktype=negotiate
#       persistent=28800
#FOR IMAP SSL
virtual=192.168.1.106:993
        real=192.168.1.104:993 gate
        fallback=192.168.1.105:993 gate
        service=imaps
        scheduler=wlc
        protocol=tcp
#FOR POP3 SSL
virtual=192.168.1.106:995
        real=192.168.1.104:995 gate
        fallback=192.168.1.105:995 gate
        service=pops
        checktype = ping
        scheduler=wlc
        protocol=tcp
```

```
#FOR MONIT MONITORING #1
virtual=192.168.1.106:10001
        real=192.168.1.104:10001 gate
        checktype = on
#FOR MONIT MONITORING #2
virtual=192.168.1.106:20001
        real=192.168.1.105:20001 gate
        checktype = on
```

virtual is the virtual IP of the services (eg 192.168.1.106 and 107)

real are the real servers IP in the cluster (192.168.1.104 and 105)

fallback is the backup server. If real IP fails then requests are forwarded to the fallback IP but they are not load balanced.

This config is based on my personal experience. Some services are load balanced, other not. Everything related to mail is not load balanced. You dont want one message to arrive on the first server and the second on the other (unless you have shared storage). If you dont have a very high traffic of mail there is no point to load balance mail (but its still highly available), the same for DNS. Later on we will rsync the messages on the second server so we will have a backup in the event that the first server fails.

About port 81. We will use it for webmail. Also I use it for our e-commerce website administration because of images upload. Later on we will set up rsync from web1.example.com to web2.example.com but not the other way around. Basically you dont want to upload a file on web2.example.com (unless you use shared storage).

If you want to get more info on the subject search for "ldirectord man".

Afterwards we create the system startup links for heartbeat and remove those of ldirectord because ldirectord will be started by the heartbeat daemon:

```
update-rc.d -f ldirectord remove
```

Finally we start heartbeat (and with it ldirectord):

```
/etc/init.d/ldirectord stop

/etc/init.d/heartbeat start
```

12.5 Test the load balancers

Let's check if both load balancers work as expected:

```
ip addr sh eth0
```

The active load balancer lb1.example.com should list the virtual IP addresses (192.168.1.106 and 192.168.1.107):

```
2: eth0:  mtu 1500 qdisc pfifo_fast qlen 1000
    link/ether 00:00:00:00:00:00 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.102/24 brd 192.168.1.255 scope global eth0
    inet 192.168.1.106/24 brd 192.168.1.255 scope global eth0
    inet 192.168.1.107/24 brd 192.168.1.255 scope global secondary eth0
```

The hot-standby (lb2.example.com) should show something like this:

```
2: eth0:  mtu 1500 qdisc pfifo_fast qlen 1000
    link/ether 00:0c:29:34:d7:7e brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.103/24 brd 192.168.1.255 scope global eth0
```

Now try :

```
ldirectord ldirectord.cf status
```

Output on the active load balancer (lb1) :

```
ldirectord for /etc/ha.d/ldirectord.cf is running with pid: 5321
```

Output on the hot standby load balancer (lb2) :

```
ldirectord is stopped for /etc/ha.d/ldirectord.cf
```

Now we will check if ports are forwarded correctly :

```
ipvsadm -L -n | grep :80
```

You should see something like this on lb1.example.com:

```
  -> RemoteAddress:Port           Forward Weight ActiveConn InActConn
TCP  192.168.1.106:80 wlc
  -> 192.168.1.104:80             Route   1      0          0
  -> 192.168.1.105:80             Route   0      0          0
TCP  192.168.1.107:80 wlc
  -> 192.168.1.104:80             Route   1      0          0
  -> 192.168.1.105:80             Route   0      0          0
```

And nothing on lb2.example.com.

One last test :

```
/etc/ha.d/resource.d/LVSSyncDaemonSwap master status
```

Output on the active load balancer:

```
master running (ipvs_syncmaster pid: 5470)
```

Output on the hot-standby:

```
master stopped
```

12.6 Testing the load balancers failover

lb1.example.com

```
/etc/init.d/heartbeat stop
```

The ipvsadm command :

```
ipvsadm -L -n | grep :80
```

should output nothing.

On lb2.example.com

```
ipvsadm -L -n | grep :80
```

and you should see the following :

```
 -> RemoteAddress:Port          Forward Weight ActiveConn InActConn
TCP  192.168.1.106:80 wlc
 -> 192.168.1.104:80            Route   1      0          0
 -> 192.168.1.105:80            Route   0      0          0
TCP  192.168.1.107:80 wlc
 -> 192.168.1.104:80            Route   1      0          0
 -> 192.168.1.105:80            Route   0      0          0
```

Restart heartbeat service on lb1.example.com :

```
/etc/init.d/heartbeat start
```

If you retry the ipvsadm command on both you will see that lb1.example.com is now active while lb2.example.com went back on standby.

If your test went fine you can go on.

13. Server Monitoring With munin And monit (web1, web2)

In this chapter I will describe how you can monitor your webservers node with munin and monit. Munin produces nifty little graphics about nearly every

aspect of your server (load average, memory usage, CPU usage, MySQL throughput, eth0 traffic, etc.) without much configuration, whereas monit checks the availability of services like Apache, MySQL, Postfix and takes the appropriate action such as a restart if it finds a service is not behaving as expected. The combination of the two gives you full monitoring: graphics that let you recognize current or upcoming problems (like "We need a bigger server soon, our load average is increasing rapidly."), and a watchdog that ensures the availability of the monitored services.

Although munin lets you monitor more than one server, we will only discuss the monitoring of the system where it is installed here.

13.1 Install And Configure munin

```
apt-get install munin munin-node
```

Next, we must edit the munin configuration file `/etc/munin/munin.conf`.

```
mv /etc/munin/munin.conf /etc/munin/munin.conf.bak

vi /etc/munin/munin.conf
```

On web1.example.com

```
dbdir   /var/lib/munin
htmldir /var/www/example/web/monitoring
logdir  /var/log/munin
rundir  /var/run/munin
tmpldir /etc/munin/templates
[web1.example.com]
    address 127.0.0.1
    use_node_name yes
```

On web2.example.com

```
dbdir   /var/lib/munin
htmldir /var/www/example/web/monitoring
logdir  /var/log/munin
rundir  /var/run/munin
tmpldir /etc/munin/templates
[web2.example.com]
   address 127.0.0.1
   use_node_name yes
```

On web1 AND web2

Next we create the directory `/var/www/example/web/monitoring` and change its ownership to the user and group munin, otherwise munin cannot place its output in that directory. Then we restart munin:

```
mkdir -p /var/www/example/web/monitoring

chown munin:munin /var/www/example/web/monitoring

/etc/init.d/munin-node restart
```

Now it is a good idea to password-protect the directory `/var/www/example/web/monitoring` unless you want everybody to be able to see every little statistic about your server.

To do this, we create an .htaccess file in `/var/www/example/web/monitoring`:

```
vi /var/www/example/web/monitoring/.htaccess
```

```
AuthType Basic
AuthName "Members Only"
AuthUserFile /var/www/example/monitoring/.htpasswd
<limit GET PUT POST>
require valid-user
</limit>
```

Then we must create the password file *`/var/www/example/.htpasswd`*. We want to log in with the username *`admin`*, so we do this:

```
htpasswd -c /var/www/example/web/monitoring/.htpasswd admin
```

Enter a password for admin, and you're done!

Now you can access reports (will take a few minutes to collect data) at these addresses :

`http://www.example.com:10001/monitoring` for web1.example.comand

`http://www.example.com:20001/monitoring` for web2.example.com.

13.2 Install And Configure monit

To install monit, we do this:

```
apt-get install monit
```

Now we must edit `/etc/monit/monitrc`. The default `/etc/monit/monitrc` has lots of examples, and you can find more configuration examples on **http://www.tildeslash.com/monit/doc/examples.php**. However, in my case I want to monitor proftpd, mysql, apache, and postfix, I want to enable the monit web interface on port 2812, I want a https web interface, I want to log in to the web interface with the username admin and the password test, and I want monit to send email alerts to root@localhost, so my file looks like this:

On web1.example.com

```
cp /etc/monit/monitrc /etc/monit/monitrc_orig

cat /dev/null > /etc/monit/monitrc

vi /etc/monit/monitrc
```

```
set daemon  60
set logfile syslog facility log_daemon
set mailserver localhost
set mail-format { from: root@web1.example.com }
set alert admin@example.com
set httpd port 2812 and
    SSL ENABLE
    PEMFILE  /var/certs/monit.pem
    allow admin:test
#check process vsftpd with pidfile /var/run/vsftpd/vsftpd.pid
#   start program = "/etc/init.d/vsftpd start"
#   stop program  = "/etc/init.d/vsftpd stop"
#   if failed host 192.168.1.104 port 21 protocol ftp then restart
#   if 5 restarts within 5 cycles then timeout
```

```
check process mysql with pidfile /var/run/mysqld/mysqld.pid
  group database
  start program = "/etc/init.d/mysql start"
  stop program = "/etc/init.d/mysql stop"
  if failed host 127.0.0.1 port 3306 then restart
  if 5 restarts within 5 cycles then timeout
check process apache with pidfile /var/run/apache2.pid
  group www
  start program = "/etc/init.d/apache2 start"
  stop program  = "/etc/init.d/apache2 stop"
  if failed host 192.168.1.104 port 80  protocol http
    and request "/example/web/monit/token" then restart
  if cpu is greater than 60% for 2 cycles then alert
  if cpu > 80% for 5 cycles then restart
  if children > 250 then restart
  if loadavg(5min) greater than 10 for 8 cycles then stop
  if 3 restarts within 5 cycles then timeout
check process postfix with pidfile /var/spool/postfix/pid/master.pid
  group mail
  start program = "/etc/init.d/postfix start"
  stop  program = "/etc/init.d/postfix stop"
  if failed port 25 protocol smtp then restart
  if 5 restarts within 5 cycles then timeout
check process named with pidfile /var/lib/named/var/run/bind/run/named.pid
  group bind
  start program = "/etc/init.d/bind9 start"
  stop  program = "/etc/init.d/bind9 stop"
  if failed port 53 then restart
  if 5 restarts within 5 cycles then timeout
```

On web2.example.com

```
cp /etc/monit/monitrc /etc/monit/monitrc_orig

cat /dev/null > /etc/monit/monitrc

vi /etc/monit/monitrc
```

```
set daemon  60
set logfile syslog facility log_daemon
set mailserver localhost
set mail-format { from: root@web2.example.com }
set alert admin@example.com
set httpd port 2812 and
    SSL ENABLE
    PEMFILE  /var/certs/monit.pem
    allow admin:test
#check process vsftpd with pidfile /var/run/vsftpd/vsftpd.pid
#   start program = "/etc/init.d/vsftpd start"
#   stop program  = "/etc/init.d/vsftpd stop"
#   if failed host 192.168.1.105 port 21 protocol ftp then restart
#   if 5 restarts within 5 cycles then timeout
check process mysql with pidfile /var/run/mysqld/mysqld.pid
   group database
   start program = "/etc/init.d/mysql start"
   stop program = "/etc/init.d/mysql stop"
   if failed host 127.0.0.1 port 3306 then restart
   if 5 restarts within 5 cycles then timeout
check process apache with pidfile /var/run/apache2.pid
   group www
   start program = "/etc/init.d/apache2 start"
   stop program  = "/etc/init.d/apache2 stop"
```

```
  if failed host 192.168.1.105 port 80  protocol http
    and request "/example/web/monit/token" then restart
  if cpu is greater than 60% for 2 cycles then alert
  if cpu > 80% for 5 cycles then restart
  if children > 250 then restart
  if loadavg(5min) greater than 10 for 8 cycles then stop
  if 3 restarts within 5 cycles then timeout
check process postfix with pidfile /var/spool/postfix/pid/master.pid
  group mail
  start program = "/etc/init.d/postfix start"
  stop  program = "/etc/init.d/postfix stop"
  if failed port 25 protocol smtp then restart
  if 5 restarts within 5 cycles then timeout
check process named with pidfile /var/lib/named/var/run/bind/run/named.pid
  group bind
  start program = "/etc/init.d/bind9 start"
  stop  program = "/etc/init.d/bind9 stop"
  if failed port 53 then restart
  if 5 restarts within 5 cycles then timeout
```

The configuration file is pretty self-explaining; if you are unsure about an option, take a look at the monit documentation: **http://www.tildeslash.com/monit/doc/manual.php**

In the apache part of the monit configuration you find this:

```
if failed host www.example.com port 80 protocol http
    and request "/example/web/monit/token" then restart
```

which means that monit tries to connect to www.example.com on port 80 and tries to access the file /monit/token which is `/var/www/example/web/monit/token` because our web site's document root is `/var/www/example/web`. If monit doesn't succeed it means Apache isn't

running, and monit is going to restart it.

On web1 AND web2

Now we must create the file `/var/www/example/web/monit/token` and write some random string into it:

```
mkdir /var/www/example/web/monit

echo "hello" > /var/www/example/web/monit/token
```

Next we create the pem cert (/var/certs/monit.pem) we need for the SSL-encrypted monit web interface:

```
mkdir /var/certs

cd /var/certs
```

We need an OpenSSL configuration file to create our certificate. It can look like this:

```
vi /var/certs/monit.cnf
```

```
# create RSA certs - Server
RANDFILE = ./openssl.rnd
[ req ]
default_bits = 1024
```

```
encrypt_key = yes
distinguished_name = req_dn
x509_extensions = cert_type
[ req_dn ]
countryName = Country Name (2 letter code)
countryName_default = MO
stateOrProvinceName             = State or Province Name (full name)
stateOrProvinceName_default     = Monitoria
localityName                    = Locality Name (eg, city)
localityName_default            = Monittown
organizationName                = Organization Name (eg, company)
organizationName_default        = Monit Inc.
organizationalUnitName          = Organizational Unit Name (eg, section)
organizationalUnitName_default  = Dept. of Monitoring Technologies
commonName                      = Common Name (FQDN of your server)
commonName_default              = server.monit.mo
emailAddress                    = Email Address
emailAddress_default            = root@monit.mo
[ cert_type ]
nsCertType = server
```

Now we create the certificate like this:

```
openssl req -new -x509 -days 365 -nodes -config ./monit.cnf -out /var/certs/monit.pem -keyout /var/certs/monit.pem
```

```
openssl gendh 512 >> /var/certs/monit.pem
```

```
openssl x509 -subject -dates -fingerprint -noout -in /var/certs/monit.pem
```

```
chmod 700 /var/certs/monit.pem
```

Afterwards we edit `/etc/default/monit` to enable the monit daemon. Change `startup` to `1` and set `CHECK_INTERVALS` to the interval in seconds that you would like monit to check your system. I choose 60 (seconds) so my file looks like this:

```
vi /etc/default/monit
```

```
# Defaults for monit initscript
# sourced by /etc/init.d/monit
# installed at /etc/default/monit by maintainer scripts
# Fredrik Steen
# You must set this variable to for monit to start
startup=1
# To change the intervals which monit should run uncomment
# and change this variable.
CHECK_INTERVALS=60
```

Finally, we can start monit:

```
/etc/init.d/monit start
```

Now point your browser to `https://192.168.1.104:2812/` or `https://192.168.1.105:2812/`, log in with admin and test, and you should see the monit web interface.

14. Mirroring web & mail files with rsync (web1, web2)

This is the tricky part. Depending on the type of website(s) you will put on the cluster, the technique used to mirror web files might be different.

We will consider web1.example.com as the master, web2.example.com will sync to web1.example.com

This can create problems in some cases. If you have a website were users can upload files, you dont want them to be uploaded on web2.example.com because the two servers will not be in sync.

In my case, I redirect all page were users can upload data to HTTP port 81 which is only forwarded to one server, web1.example.com. If the master server fails, then all port 81 traffic is forwarded to the fallback server, web2.example.com and mirroring is temporary stopped.

Redirect can be done with in apache vhost.conf :

```
[...]
Redirect /admin http://www.example.com:81/webmail
[...]
```

If you have a lot of file upload done by users, you might consider using a load balance NFS :

http://www.howtoforge.com/high_availability_nfs_drbd_heartbeat

14.1 Install rsync

```
apt-get install rsync
```

14.2 Create The Keys On web2.example.com

Now we create the private/public key pair on web2.example.com:

```
mkdir /root/rsync

ssh-keygen -t dsa -b 1024 -f /root/rsync/mirror-rsync-key
```

You will see something like this:

```
Generating public/private dsa key pair.
Enter passphrase (empty for no passphrase): [press enter here]
Enter same passphrase again: [press enter here]
Your identification has been saved in /root/rsync/mirror-rsync-key.
Your public key has been saved in /root/rsync/mirror-rsync-key.pub.
The key fingerprint is:
68:95:35:44:91:f1:45:a4:af:3f:69:2a:ea:c5:4e:d7 root@web2
```

It is important that you do not enter a passphrase otherwise the mirroring will not work without human interaction so simply hit enter!

Next, we copy our public key to web1.example.com:

```
scp /root/rsync/mirror-rsync-key.pub vmail@web1.example.com:/home/vmail/
```

The public key mirror-rsync-key.pub should now be available in `/home/vmail` on web1.example.com.

14.3 Configure web1.example.com

web1.example.com

Now log in through SSH as vmail (not root!) and do this:

```
login vmail
```

```
mkdir ~/.ssh

chmod 700 ~/.ssh

mv ~/mirror-rsync-key.pub ~/.ssh/

cd ~/.ssh

touch authorized_keys

chmod 600 authorized_keys

cat mirror-rsync-key.pub >> authorized_keys
```

By doing this, we have appended the contents of `mirror-rsync-key.pub` to the file `/home/vmail/.ssh/authorized_keys`. `/home/vmail/.ssh/authorized_keys` should look similar to this:

(Still as webmaster! on web1.example.com)

```
vi ~/.ssh/authorized_keys
```

```
ssh-dss AAAAB3NzaC1kc3MAAA[...]lSUom root@web2
```

Now we want to allow connections only from web2.example.com, and the connecting user should be allowed to use only rsync, so we add

```
command="/home/vmail/rsync/checkrsync",from="192.168.0.105",no-port-forwarding,no-X11-forwarding,no-pty
```

right at the beginning of `/home/vmail/.ssh/authorized_keys`:

```
command="/home/vmail/rsync/checkrsync",from="192.168.0.105",no-port-forwarding,no-X11-forwarding,no-pty ssh-dss AAAAB3NzaC1kc3MAAA[...]lSUom root@
web2
```

Now we create the script `/home/vmail/rsync/checkrsync` that rejects all commands except rsync on web1.example.com.

(We still do this as vmail!)

```
mkdir ~/rsync

vi ~/rsync/checkrsync
```

```
#!/bin/sh
```

```
case "$SSH_ORIGINAL_COMMAND" in
	*\&*)
		echo "Rejected"
		;;
	*\(*)
		echo "Rejected"
		;;
	*\{*)
		echo "Rejected"
		;;
	*\;*)
		echo "Rejected"
		;;
	*\<*)
		echo "Rejected"
		;;
	*\`*)
		echo "Rejected"
		;;
	rsync\ --server*)
		$SSH_ORIGINAL_COMMAND
		;;
	*)
		echo "Rejected"
		;;
esac
```

```
chmod 700 ~/rsync/checkrsync
```

14.4 Test automated rsync

web2.example.com

First we have to tell rsync which file we dont want to sync (such as munit graph!)

```
vi /root/exclude_www.txt
```

and make it look like this :

```
/example/web/monit/
/example/web/monitoring/
/example/ssl/
/yoursite/ssl/
```

and now for mail :

```
vi /root/exclude_mail.txt
```

```
/rsync/
/.ssh/
#if you use my monitoring script, see next page
server1_was_down/
.bash_history/
.bash_logout/
.bashrc/
.mysql_history/
```

```
.profile/
.viminfo/
```

Now we must test on web2.example.com if we can mirror web1.example.com without being prompted for vmail's password. We do this:

(We do this as root!)

web2.example.com

```
rsync -avz --delete --ignore-errors --exclude-from '/root/exclude_www.txt' -e "ssh -i /root/rsync/mirror-rsync-key"
vmail@192.168.0.104:/var/www/ /var/www/
```

followed by :

```
rsync -avz --delete --ignore-errors --exclude-from '/root/exclude_mail.txt' -e "ssh -i /root/rsync/mirror-rsync-key"
vmail@192.168.0.104:/home/vmail/ /home/vmail/
```

You should now see that the mirroring takes place without being prompted for a password :

```
receiving file list ... done
sent 71 bytes  received 643 bytes  476.00 bytes/sec
total size is 64657  speedup is 90.56
```

14.5 Create a cron job

First we will create a script that will make sure that if for some reason web1.example.com went down, rsync will be stopped and that root as to restart manually the process.

The reason why we have to do this is because web2.example.com will take over web1.example.com for mail if it went down. New mail will then be received on web2.example.com during that period, so let say web1.example.com comes back online one hour after, all new mail during that period of time would be deleted on web2.example.com because of rsync --delete command. We will see below how to sync back both servers it that happend.

For now lets write that script.

On web2.example.com

```
vi /root/rsync_web1
```

```
#!/bin/bash
# Script to rsync web data and mail between 2 load balanced server
# Copyright (c) 2008 blogama.org
# This script is licensed under GNU GPL version 2.0 or above
# -------------------------------------------------------------------------
### This script has 2 purpose ###
### 1) Check connection on port 25 to master server and then rsync mail ###
### 2) Check connection on port 80 to master server and then rsync www data ###
### To be modified ###
MASTERSERVERIP="192.168.0.104"
WEBPORT="80"
SMTPPORT="25"
SSHPORT="22"
EMAIL="admin@example.com"
WWWRSYNCUSER="vmail"
WWWDIR="/var/www/"
MAILRSYNCUSER="vmail"
MAILDIR="/home/vmail/"
```

```
###### Do not make modifications below ######
### Binaries ###
MAIL=$(which mail)
TELNET=$(which telnet)
RSYNC=$(which rsync)
SSH=$(which ssh)
### To restore to original when problem fixed ###
if [ $1 ]; then
  if [ $1=="fix" ]; then
    if [ -f smtp_down.txt ]; then
      rm /root/smtp_down.txt
    fi
    if [ -f www_down.txt ]; then
      rm /root/www_down.txt
    fi
  fi
fi
###################SMTP###################
### If already notified for SMTP problem exit ###
cd /root
if [ -f smtp_down.txt ]; then
  exit 1;
fi
### Check if server 1 is responding on SMTP. If yes rsync mail ###
### If server 1 was down, cannot rsync --delete server 2 with 1 ###
### Must resync server1 with 2 (no delete) and run /root/sync fix ###
(
echo "quit"
) | $TELNET $MASTERSERVERIP $SMTPPORT | grep Connected > /dev/null 2>&1
if [ "$?" -ne "1" ]; then
  $RSYNC -avz --delete --ignore-errors --exclude-from '/root/exclude_mail.txt' -e "ssh -p $SSHPORT -i /root/rsync/mirror-rsync-key" $MAILRSYNCUSER@$MASTERSERVERIP:$MAILDIR $MAILDIR
else
```

```
  echo "Server 1 down. Mail sync is not working anymore" > /root/smtp_down.txt
  $MAIL -s "Server 1 down port 25" $EMAIL < /root/smtp_down.txt
fi
####################HTTP####################
### If already notified for HTTP problem exit ###
cd /root
if [ -f http_down.txt ]; then
  exit 1;
fi
### Check if server 1 is responding on HTTP. If yes rsync www data ###
(
echo "quit"
) | $TELNET $MASTERSERVERIP $WEBPORT | grep Connected > /dev/null 2>&1
if [ "$?" -ne "1" ]; then
  $RSYNC -avz --delete --ignore-errors --exclude-from '/root/exclude_www.txt' -e "ssh -p $SSHPORT -i /root/rsync/mirror-rsync-key" $WWWRSYNCUSER@$MASTERSERVERIP:$WWWDIR $WWWDIR
else
  echo "Server 1 down. WWW sync is not working anymore" > /root/http_down.txt
  $MAIL -s "Server 1 down port 80" $EMAIL < /root/http_down.txt
fi
```

And make this file executable :

```
chmod +x /root/rsync_web1
```

Of course you can put whatever you want in those files ;) If you dont put slash it will ignore the keyword recursively (ex : *.gz), if you put slash it is for a precise location (ex : `/example/ssl/`).

Now we add the script to our crontab by doing the following :

```
crontab -e
```

and add this line :

```
[...]
*/5 * * * * /root/rsync_web1  >/dev/null 2>&1
[...]
```

This will run it every 5 minutes, of course you can change that to your needs.

14.6 What if web1.example.com was down

Generally speaking, if web1.example.com is down, you then have to :

1) rsync WEB1 TO WEB2 without --delete command :

web1.example.com

```
rsync -avz vmail@192.168.0.104:/var/www/ /var/www/

rsync -avz vmail@192.168.0.104:/home/vmail/ /home/vmail/
```

15. Custom scripts for monitoring (lb1, lb2, web1, web2)

I made a few bash script to monitor the whole setup (they are a bit ugly but they work). If you make them better, feel free to mail them to me!

15.1 Monitoring from lb1.example.com

First we must install sendmail so lb1.example.com will be able to send mail :

```
apt-get install sendmail
```

The first script will check if the backup load balancer (lb2.example.com) is still available to takeover :

```
vi /root/lb2_check
```

```
#!/bin/bash
# Backup load balancer check
# Copyright (c) 2008 blogama.org
# This script is licensed under GNU GPL version 2.0 or above
# -------------------------------------------------------------------------
### This script does 1 verification ###
### 1) Check if backup load balancer failed and send mail notification ###
### To be modified ###
EMAIL="admin@example.com"
###### Do not make modifications below ######
### Binaries ###
MAIL=$(which mail)
### To restore to original when problem fixed ###
if [ $1 ]; then
  if [ $1=="fix" ]; then
    rm /root/lb2_problem.txt
    > /var/log/ha-log
    exit 1;
  fi
fi
```

```
### Check if already notified ###
cd /root
if [ -f lb2_problem.txt ]; then
  exit 1;
fi
### Check if Heartbeat is running on hot standby ###
tail /var/log/ha-log 2>&1 | grep "Asking other side for ping node count"
if [ "$?" -ne "1" ]; then
  echo "Backup load balancer failed" > /root/lb2_problem.txt
  $MAIL -s "Backup load balancer problem" $EMAIL < /root/lb2_problem.txt
fi
```

We make this script executable :

```
chmod +x /root/lb2_check
```

If the lb2.example.com fails, then it will create a file `/root/lb2_problem.txt` and send a mail notification. Until the file `lb2_problem.txt` is there, it won't check again. Also we must empty the log file once the problem is fixed for the script to work properly.

Once the problem is fixed on lb2.example.com, please manually run :

```
/root/lb2_check fix
```

The next script will check if any ports failed on either web1 or web2 by checking the ldirectord log file. There is already a mail notification with ldirectord but it sends millions of notification, mine only send one until you fix the problem :

```
vi /root/ports_failed
```

and make it look like this :

```
#!/bin/bash
# Ldirectord ports failure check
# Copyright (c) 2008 blogama.org
# This script is licensed under GNU GPL version 2.0 or above
# -------------------------------------------------------------------------
### This script does 1 verification ###
### 1) Check for port failure on load balanced servers ###
### To be modified ###
EMAIL="admin@example.com"
###### Do not make modifications below ######
### Binaries ###
MAIL=$(which mail)

#to restore to original when problem fixed
if [ $1 ]; then
  if [ $1=="fix" ]; then
    rm /root/port_problem.txt
    > /var/log/ldirectord.log
  fi
fi
###check if already notified###
cd /root
if [ -f port_problem.txt ]; then
  cat /var/log/ldirectord.log | grep Deleted > /var/log/port_problem.log
  exit 1;
fi
```

```
### Check if port failed ###
cat /var/log/ldirectord.log 2>&1 | grep Deleted
if [ "$?" -ne "1" ]; then
  cat /var/log/ldirectord.log | grep Deleted > /var/log/port_problem.log
  cat "Ports problem see logfile /var/log/port_problem.log" > /root/port_problem.txt
  $MAIL -s "Some ports failed" $EMAIL < /root/port_problem.txt
fi
```

We make it executable :

```
chmod +x /root/ports_failed
```

This is the same as the first script, once the problem is fixed you must run :

```
/root/ports_failed fix
```

in order to make the script running again.

Now add both scripts to your crontab :

```
crontab -e
```

```
* * * * * /root/ports_failed  >/dev/null 2>&1
* * * * * /root/lb2_check  >/dev/null 2>&1
```

15.2 Monitoring from lb2.example.com

Monitoring the second load balancer is important because it will tell us if the master load balancer failed and if it did, keep an eye for ports failure on web1 and web2.

First we must install sendmail so lb2.example.com will be able to send mail :

```
apt-get install sendmail
```

```
vi /root/ports_check
```

And paste this script :

```
#!/bin/bash
# Ldirectord ports failure check
# Copyright (c) 2008 blogama.org
# This script is licensed under GNU GPL version 2.0 or above
# -------------------------------------------------------------------

### This script does 2 verifications ###
### 1) check if master load balancer failed and send mail notification ###
### 2) If master load balancer failed, check for port failure on load balanced servers ###
### To be modified ###
EMAIL="admin@example.com"
###### Do not make modifications below ######
### Binaries ###
MAIL=$(which mail)
### Date ###
```

```
NOW=$(date)
### To restore to original when problem fixed ###
if [ $1 ]; then
  cd /root/
  if [ $1=="fix" ]; then

    if [ -f lb1_problem.txt ]; then
      rm /root/lb1_problem.txt
    fi

    if [ -f port_problem.txt ]; then
      rm /root/port_problem.txt
    fi

    if [ -f /root/server_problem_notified.txt ]; then
      rm /root/server_problem_notified.txt
    fi
  > /var/log/ldirectord.log
  > /var/log/ha-log
  exit 1;
  fi
fi
#check if ldirectord is running on lb2.example.com (means that lb1.example.com failed)
#$LDIRECTORD /etc/ha.d/ldirectord.cf status 2>&1 | grep running
cat /var/log/ha-log | grep "takeover complete" > /dev/null 2>&1
if [ "$?" -ne "1" ]; then
  ###check if already notified###
  cd /root
  if [ -f port_problem.txt ]; then
    cat /var/log/ldirectord.log | grep Deleted > /var/log/port_problem.log
    exit 1;
  fi
  ### Check if port failed ###
```

```
  cat /var/log/ldirectord.log 2>&1 | grep Deleted
  if [ "$?" -ne "1" ]; then
    cat /var/log/ldirectord.log | grep Deleted > /var/log/port_problem.log
    echo "Ports problem see logfile /var/log/port_problem.log" > /root/port_problem.txt
    $MAIL -s "Some ports failed" $EMAIL < /root/port_problem.txt
  fi

  ### Check if already notified that master load balancer failed ###
  cd /root
  if [ -f server_problem_notified.txt ]; then
    exit 1;
  fi

  ### Notify that master load balancer failed ###
  cd /root
  MESSAGE="$NOW : Master load balancer failed"
  echo $MESSAGE > lb1_problem.txt
  $MAIL -s "Master load balancer failed" $EMAIL < /root/lb1_problem.txt
  echo "notified" > server_problem_notified.txt
fi
```

We make it executable :

```
chmod +x /root/ports_check
```

And we add it to our crontab :

```
crontab -e
```

```
* * * * * /root/ports_failed  >/dev/null 2>&1
```

When you get a notification from the script, please run afterward :

```
/root/ports_check fix
```

15.3 Monitoring from web1 & web2

Monitoring of web cluster is already partially done with monit and munin.

The part that is not covered yet is the monitoring of MySQL replication.

Please read the following article : **Repair MySQL master-master replication**

MySQL monitoring is optional but on a production server, problems can happend with MySQL replication so I really recommend using those scripts or something similar to check databases consistency.

15.4 Monitoring from remote server

This part is adding extra security by checking important ports (25,53,80,443) from a remote server (install dns-utils for dig):

```
#!/bin/bash
# Script to check important port on remote webserver
# Copyright (c) 2008 blogama.org
# This script is licensed under GNU GPL version 2.0 or above
# -------------------------------------------------------------------
### This script does a verification on port 25, 53, 80 and 443 ###
### After 2 failed check it will send a mail notification ###
### To be modified ###
```

```
WEBSERVERIP="192.168.1.106"
MAILSERVERIP="192.168.1.106"
EMAIL="admin@example.com"
DNSSERVERIP="192.168.1.106"
DOMAINTOCHECKDNS="example.com"
DOMAINIP="192.168.1.106"

###### Do not make modifications below ######
### Binaries ###
MAIL=$(which mail)
TELNET=$(which telnet)
DIG=$(which dig)
### Check if already notified###
cd /root
if [ -f server_problem.txt ]; then
  exit 1;
fi
### Test SMTP ###
(
echo "quit"
) | $TELNET $MAILSERVERIP 25 | grep Connected > /dev/null 2>&1
if [ "$?" -ne "1" ]; then
  echo "PORT CONNECTED"
else
  if [ -f server_problem_first_time_25.txt ]; then
    echo "PORT 25 NOT CONNECTED" >> /root/server_problem.txt
  else
    echo "NOT CONNECTED" > /root/server_problem_first_time_25.txt
  fi
fi
### Test HTTP ###
(
echo "quit"
```

```
) | $TELNET $WEBSERVERIP 80 | grep Connected > /dev/null 2>&1
if [ "$?" -ne "1" ]; then
  echo "PORT CONNECTED"
else
  if [ -f server_problem_first_time_80.txt ]; then
    echo "PORT 80 NOT CONNECTED" >> /root/server_problem.txt
  else
    echo "NOT CONNECTED" > /root/server_problem_first_time_80.txt
  fi
fi
### Test HTTPS###
(
echo "quit"
) | $TELNET $WEBSERVERIP 443 | grep Connected > /dev/null 2>&1
if [ "$?" -ne "1" ]; then
  echo "PORT CONNECTED"
else
  if [ -f server_problem_first_time_443.txt ]; then
    echo "PORT 81 NOT CONNECTED" >> /root/server_problem.txt
  else
    echo "NOT CONNECTED" > /root/server_problem_first_time_443.txt
  fi
fi

### Test DNS ###
$DIG $DOMAINTOCHECKDNS @$DNSSERVERIP | grep $DOMAINIP
if [ "$?" -ne "1" ]; then
  echo "PORT CONNECTED"
else
  if [ -f server_problem_first_time_53.txt ]; then
    echo "PORT 53 NOT CONNECTED" >> /root/server_problem.txt
  else
    echo "NOT CONNECTED" > /root/server_problem_first_time_53.txt
```

```
  fi
fi
### Send mail notification after 2 failed check ###
if [ -f server_problem.txt ]; then
  $MAIL -s "Server problem" $EMAIL < /root/server_problem.txt
fi
```

Et voila! Feel free to send me private emails at admin [at] marchost.com or post comments here or on my page : **blogama.org**