Redmine Installation on Debian

v1.1

Introduction

1. Objectives

Have a fully functional Redmine installation on a dedicated server with good performance.

The idea of this document came after an easy installation that use fastcgi and was nor easy nor fast! So I decided to write this documentation about how to be able to install Redmine from the beginning to the very end.

2. Presentation of the Document

As this document describes the Redmine installation process for a specific environment, it is divided into two parts. The first part gives the basic clues for a fully functional installation. The second part details the action taken to tune the server and adds some backup facilities.

3. Choices

Technology	Details
Linux – Debian	Stability!
MySQL	The database recommended for Redmine
Apache2	Standard and robust web server
Passenger	Integration of Rails into Apache with good performance
Subversion	The source control used for my projects

Summary

Installation Steps

1. Initial Download

There is only one initial download for this installation:

- Debian “Lenny” 5.0.1 - debian-501-i386-businesscard.iso

2. Install Debian

The Debian server is installed with the classic installation process and without any special commands or actions. This way we could have quickly a functional environment and tune it afterward.

The first questions depend of your own regional settings, your own network settings and your own installation procedure. Your choices have no impact on the Redmine installation process.

For your information, here are my choices:

- Language: English
- Country: Luxembourg
- Keyboard map: French
- Hostname: Debian
- Domain: Belkin
- Mirror: ftp.debian.skynet.be
- Proxy: without proxy
- Partionning: entire disk, one primary partition
- Root password: *****
- User name: debian
- User login: debian
- User password: *****
- Popularity Context: No

The first question that matter is the one about the packages to install. You need to select the options “Web Server” and “Standard system”. All others are optional and depend of your own procedures (Don’t hesitate to ignore them).

For your information, here are the soft installed by those packages:

- Desktop environment: gnome
- Web server: apache2
- Print server: cups
- DNS server: bind9
- File server: samba, nfs
- Mail server: exim4, spamassassin, uw-imap
- SQL database: postgresql
- Laptop: bluetooth, hibernate, …
- Standard system: ftp, telnet, whois, doc-debian, …

Finalize the wizard.

Eject. Reboot. First step finished! You have a Debian server.

More Information: http://www.debian.org/releases/stable/i386/ch06s03.html

3. Install and Configure MySQL

Debian provides a standard "mysql-server" package that will install the version 5.0.51a-24.

```
aptitude install mysql-server
```

The default data path is "/var/lib/mysql" and the configuration can be found in "/etc/mysql".

The next step is to create the initial database and the associated user:

```
mysql -p
create database redmine character set utf8;
grant all on redmine.* to 'redmine' identified by '*****';
```

As the default storage engine is MyISAM and as I want a system that I can rely on (with transactions and foreign keys in fact): I switch the default storage engine to InnoDB. This step is optional, but essential in my point-of-view.

For this, I create a file "/etc/mysql/conf.d/innodb.cnf" that contains the following lines:

```
[mysqld]
default-storage-engine = InnoDB
```

Do it if – as me – you prefer InnoDB to MyISAM and then restart the server:

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

4. Install Ruby Packages

The Redmine 0.8.x requirements are:

- Ruby 1.8.6 or 1.8.7,
- Rails 2.1.2,
- RubyGems 1.3.1,
- Rake 0.8.3

The associated Debian packages are:

Package	Version	Details
ruby	1.8.7.72-3	
rails	2.1.0-6	Dependency to ruby and rake
rubygems	1.2.0-3	Dependency to ruby
rake	0.8.1-3	Dependency to ruby

It's not exactly the mandatory versions but the most important is to install the good version of Rails. For this reason, Rails will be installed with gem and not with apt.

```
aptitude install ruby rake rubygems
aptitude install libmysql-ruby libopenssl-ruby
gem install rails –v=2.1.2
```

5. Install Passenger (aka mod_rails)

To install Passenger, we will have to compile it from its source code as there is no existing package for Debian. So we need a compiler and the mandatory libraries.

```
aptitude install g++ make ruby-dev apache2-dev
gem install passenger
cd /var/lib/gems/1.8/gems/passenger-2.2.1/bin
./passenger-install-apache2-module
```

As the path of passenger is a little bit complicated, I decide to create a link into /usr/local/lib. A second option could be a movement of all the files as described in the Passenger documentation (http://www.modrails.com/documentation/Users%20guide.html#_moving_phusion_passenger_to_a_different_directory).

```
ln -s /var/lib/gems/1.8/gems/passenger-2.2.1 /usr/local/lib/passenger
```

The Passenger installer gives some instructions to activate the mod into the Apache configuration. To respect the current Apache architecture, the deployment of the new mod configuration will be done by creating two files: passenger.conf and passenger.load in the "/etc/apache2/mods-available" directory.

passenger.load

```
LoadModule passenger_module
/usr/local/lib/passenger/ext/apache2/mod_passenger.so
```

passenger.conf

```
PassengerRoot /usr/local/lib/passenger-2.2.1
PassengerRuby /usr/bin/ruby1.8
PassengerDefaultUser www-data
```

To activate the mod, a link needs to be created into the "mods-enabled" directory.

```
cd /etc/apache2/mods-enabled
ln –s ../mods-available/passenger.load
ln –s ../mods-available/passenger.conf
/etc/init.d/apache2 reload
```

More information: http://modrails.com/documentation/Users%20guide.html

6. Install Redmine

By downloading Redmine from its subversion repository, you could be easily deployed the future "point releases".

```
aptitude install subversion
```

```
cd /usr/local/lib
svn co http://redmine.rubyforge.org/svn/branches/0.8-stable redmine-0.8
```

Next, you need to connect Redmine to its database by creating a database.yml file.

```
cd /usr/local/lib/redmine-0.8
cp config/database.yml.example config/database.yml
vi config/database.yml
```

The default configuration uses the root user without password. You need to configure the "production" information in the database.yml file to have a configuration like this one:

```
...
production:
  adapter: mysql
  database: redmine
  host: localhost
  username: redmine
  password: *****
  encoding: utf8
...
```

Now you can populate the database with its default tables and data.

```
rake db:migrate RAILS_ENV="production"
rake redmine:load_default_data RAILS_ENV="production"
```

The selection of a language is only a default choice as all the language will be installed, and you will be able to change your mind latter via the administration of Redmine.

Source: http://www.redmine.org/wiki/redmine/RedmineInstall

7. Configure Apache

As the user that runs both Apache and Passenger is www-data, it requires some rights on the Redmine web site folders.

```
cd /usr/local/lib/redmine-0.8
mkdir tmp public/plugin_assets
chown -R rwww-data:www-data files log tmp public/plugin_assets
chmod -R 755 files log tmp public/plugin_assets
```

Finally, configure Apache to work with this web site by creating a file "/etc/apache2/sites-available/redmine" with the following content:

```
<VirtualHost *:80>
    DocumentRoot /usr/local/lib/redmine-0.8/public
    <Directory /usr/local/lib/redmine-0.8/public>
        AllowOverride None
    </Directory>
</VirtualHost>
```

To enable this site, create a link into the directory "sites-enabled".

```
cd /etc/apache2/sites-enabled
rm 000-default
ln –s ../sites-available/redmine 000-redmine
/etc/init.d/apache2 reload
```

Done! Launch your favorite browser and enjoy Redmine!

8. Subversion (svnserve)

To use svnserve as a daemon, the first step is to create the associated script in "/etc/init.d".

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

After the edition of this file, we will be able to have a fully functional Subversion server easily. Here are the values for the modified variables of the script:

```
DESC="Subversion server"
NAME=svnserve
DAEMON=/usr/bin/$NAME
DEAMON_ARGS="-d –r /var/local/svn"
```

Now, the daemon needs to be deployed on the rc.xx.d directories.

```
update-rc.d svnserve defaults
```

Tuning Steps

1. Apache

The configuration done just before is really light.

Here is a sample of a much better configuration for the file "/etc/apache2/sites-available/redmine":

```
<VirtualHost *:80>
    DocumentRoot /usr/local/lib/redmine-0.8/public
    ServerSignature off

    <Directory />
        Order Deny,Allow
        Deny from all
    </Directory>

    <Directory /usr/local/lib/redmine-0.8/public>
        AllowOverride None
        Order allow,deny
        Allow from all
    </Directory>

    ErrorLog /var/log/apache2/redmine-error.log
    CustomLog /var/log/apache2/redmine-access.log combined
</VirtualHost>
```

More information: http://httpd.apache.org/docs/2.2/

2. Passenger

The configuration for Passenger could be improved by reducing a DoS risk and by globalize the queue of the Passenger threads.

For this, the new "/etc/apache2/mods-available/passenger.conf" configuration file is:

```
<IfModule mod_passenger.c>

PassengerRoot /usr/local/lib/passenger-2.2.1
PassengerRuby /usr/bin/ruby1.8
PassengerDefaultUser www-data
PassengerUserSwitching off
PassengerUseGlobalQueue on
PassengerRestartDir /var/local/passenger

</IfModule>
```

And we need to create the associated "restart.txt" file.

```
mkdir /var/local/passenger
> /var/local/passenger/restart.txt
```

More information: http://www.modrails.com/documentation/Users%20guide.html

3. Subversion

To simply the creation of new projects on Subversion, I have created a “svn-create-project.sh” script in “/usr/local/bin/” that does the following actions:

- Create the new project in the Subversion repository,
- Prepare its configuration and use a unique password file,
- Initialize its directory hierarchy with branches, tags and trunk folders,
- Create a post-commit hook that updates Redmine on each commit,
- Create a pre-commit hook that checks that each commit have a message.

/usr/local/bin/svn-create-project.sh

```
#!/bin/sh

if [ “$1” == “” ]
then
  echo “missing project name”
else
  SVN_REP=/var/local/svn
  svnadmin create “$SVN_REP/$1”
  cp “$SVN_REP/.config/svnserve.conf” “$SVN_REP/$1/conf”
  cp “$SVN_REP/.config/hooks/”* “$SVN_REP/$1/hooks”
  rm “$SVN_REP/$1/conf/passwd”
  svn import “$SVN_REP/.config/initial” file://$SVN_REP/$1 --message
“Initial directories”
fi
```

As you can see this script use a special directory (“.config”) created into the Subversion repository. This directory is created via the following commands:

```
mkdir /var/local/svn/.config
mkdir /var/local/svn/.config/initial
mkdir /var/local/svn/.config/initial/branches
mkdir /var/local/svn/.config/initial/tags
mkdir /var/local/svn/.config/initial/trunk
mkdir /var/local/svn/.config/hooks
```

This special directory contains the following 2 files: the “passwd” that are common to all projects and the “svnserve.conf” that will be used to initialize all the projects.

/var/local/svn/.config/passwd

```
[users]
redmine = *****
francois = *****
```

/var/local/svn/.config/svnserve.conf

```
[general]
password-db = ../../.config/passwd
```

Finally, the .config/hook directory contains a post-commit and a pre-commit hooks.

/var/local/svn/.config/hooks/post-commit

```
#!/bin/sh

REPOS="$1"
REV="$2"
REDMINE="/usr/local/lib/redmine-0.8"

/usr/bin/ruby $REDMINE/script/runner "Repository.fetch_changesets" -e
production
```

/var/local/svn/.config/hooks/pre-commit

```
#!/bin/sh

REPOS="$1"
TXN="$2"

/usr/bin/svnlook log -t "$TXN" "$REPOS" | \
    grep "[a-zA-Z0-9]" > /dev/null || exit 1

exit 0
```

As those scripts will be executed, don't forget to give it the executable right.

```
chmod +x post-commit
chmod +x pre-commit
```

Backup

1. Preparation

The idea is to create a full backup each day with an incremental backup each hour. The proposed system is really simple because I don't want to go to high availability (and create a slave server) and because my directory "/backup" could be itself secured by other tools.

To prepare my backup scripts, I need rsync but it is not part of the default installation of Debian.

```
aptitude install rsync
```

Now, I need to create the directory hierarchy for the backup.

```
mkdir /backup
mkdir /backup/redmine
mkdir /backup/redmine/files
mkdir /backup/redmine/mysql
```

2. MySQL Configuration

MySQL needs to be configured because incremental backup is available on MySQL only if the "binary logs" is enabled.

For this, create an “etc/mysql/conf.d/backup.cnf” file with the following lines:

```
[mysqld]
log_bin=redmine-bin
expire_logs_days=2
```

And restart the server.

```
/etc/init.d/mysqld restart
```

Source: http://dev.mysql.com/doc/mysql-security-excerpt/5.0/en/backup.html

3. Full Backup

For the full backup script, I create a file “/usr/local/bin/redmine-backup-full.sh” with the following code:

```
#!/bin/sh

# Redmine Attachments
rsync -a /usr/local/lib/redmine-0.8/files /backup/redmine/files

# MySQL Extract
mysqldump –password=**** --single-transaction --quick –-flush-logs --
master-data=2 --delete-master-logs redmine | gzip >
/backup/redmine/mysql/`date +%Y%m%d`.gz
rm –f /backup/redmine/mysql/redmine-bin*
```

The file needs to be registered as an executable.

```
chmod +x /usr/local/bin/redmine-backup-full.sh
```

4. Incremental Backup

For the incremental backup script, I create a file “/usr/local/bin/redmine-backup-incremental.sh” with the following code:

```
#!/bin/sh

# Redmine Attachments
rsync -a /usr/local/lib/redmine-0.8/files /backup/redmine/files

# MySQL Incremental Backup
rsync –a /var/lib/mysql/redmine-bin* /backup/redmine/mysql
```

The file needs to be registered as an executable.

```
chmod +x /usr/local/bin/redmine-backup-incremental.sh
```

5. Cron

Last thing to do: schedule the backup tasks. Without any surprise, we do this with cron. The standard command for editing cron scheduling is:

```
crontab –e
```

And here is the schedule data for those tasks:

```
# min hour dom mon dow   command
    0    1   *   *   *   redmine-backup-full.sh
   30    *   *   *   *   redmine-backup-incremental.sh
```