|  |  |
| --- | --- |
|  | **2014** |
|  | The Company  Martien Huijsmans |

|  |
| --- |
| **[RPM]** |
| todo |

Inhoud

[Creating an RPM 1](#_Toc405743885)

[Basics of building a RPM package 1](#_Toc405743886)

[Filesystem Hierarchy Standards 1](#_Toc405743887)

[Process 1](#_Toc405743888)

[Operational procedures 2](#_Toc405743889)

[RPM properties 2](#_Toc405743890)

[RPM tooling 2](#_Toc405743891)

[Prepare Centos to create RPM’s 3](#_Toc405743892)

[Setting up own RPM Server 3](#_Toc405743893)

[Caching and timers 4](#_Toc405743894)

[Copying from Jenkins to repo server 4](#_Toc405743895)

[YUM commands 4](#_Toc405743896)

[Installing unsiged RPM’s 5](#_Toc405743897)

[RPM Groups 5](#_Toc405743898)

[RPM Commands 5](#_Toc405743899)

[Commands used in RPM scripts 5](#_Toc405743900)

# Creating an RPM

## Basics of building a RPM package

Next ref was the first time that a building a RPM started made sense to me.  
Ref: <https://fedoraproject.org/wiki/How_to_create_an_RPM_package#The_basics_of_building_RPM_packages>

## Filesystem Hierarchy Standards

<http://www.pathname.com/fhs/>   
Site states: *The filesystem standard has been designed to be used by Unix distribution developers, package developers, and system implementors. However, it is primarily intended to be a reference and is not a tutorial on how to manage a Unix filesystem or directory hierarchy.*

# Process

Process

* identify RPM
* Process (automation, testing) needs to be such that within a day a new patch can be delivered

Architecture strategies

* Strategy for install / upgrade needed.
* Only RPM’s are installed on the system
* A delivery is a tgz, with a set of RPM’s
* 3rd party RPM are installed without modification.  
  There is a separate config RPM to configure a 3rd party installation  
  Config RPM has 3rd party RPM as dependency
* Each RPM has a test RPM.
* Use softlinks ?

# Operational procedures

Procedures needed for

* maiden install

Possible upgrade project

* systems sync to disk, e.g. DB syncs all data
* make snapshots of the logical volumes
* install new software
* on failure, restore the LVM.
  + Who does that?

# RPM properties

Requires an RPM repo

RPM has install & remove script. But an install script needs to know is something is an upgrade.

Script Technology: python

RPM needs to own (user:group) the commands (e.g. mkdir) executed by the (install/uninstall) scripts.

RPM standard: 4 & 5.

# RPM tooling & Guidelines

YUM: for RPM management

Viewing / Browsing / Listing ?

Creation of RPM’s?  
Linux tool and process explained  
<http://javaworkshop.wordpress.com/2008/10/>

RPMlint & Gudelines  
<http://www.redhat.com/promo/summit/2008/downloads/pdf/Wednesday_130pm_Tom_Callaway_OSS.pdf>

# Prepare Centos to create RPM’s

<http://wiki.centos.org/HowTos/SetupRpmBuildEnvironment>

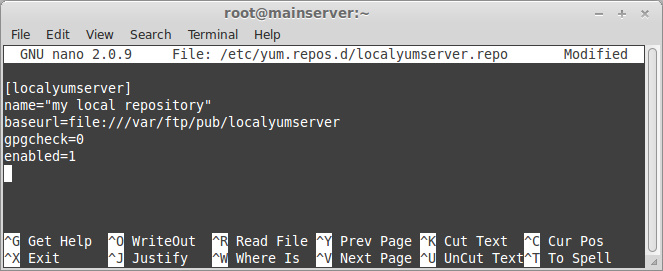
# Setting up own RPM Server

Setting up own RPM server  
<http://ostechnix.wordpress.com/2013/01/05/setup-local-yum-server-in-centos-6-x-rhel-6-x-scientific-linux-6-x/>

Install vsftpd  
$ sudo yum install vsftpd

Start vsftpd service  
$ sudo service vsftpd start

Install createrepo package if it is not installed. This is package is used to create our local repository.  
$ sudo yum install createrepo

Create a folder called localyumserver in /var/ftp/pub directory to hold all packages  
$ sudo mkdir /var/ftp/pub/localyumserver  
  
Create a repo file called localyumserver.repo in /etc/yum.repos.d/ directory  
$ sudo vi /etc/yum.repos.d/localyumserver.repo with following content  


Now it is time to create our repository. Enter the following command in the Terminal:  
$ sudo createrepo -v /var/ftp/pub/localyumserver

If you change the content of the /etc/yum.repos.d/ directory, update the yum repo (cache?)  
$ sudo yum clean all  
$ sudo yum update  
  
To run own YumServer it is probably best to remove all other .repo files, because “yum update” does a system update of all packages.  
If there is only one repo (e.g. localyumserver) the system is updated with the latest packages.

# Caching and timers

Yum maintains a cache of data downloaded from repo. It also has a timer for downloaded packages. If timer has not expired, to will not check the repo for updates.  
The parameter metadata\_expire in /etc/yum.conf can be set to control the value (default 1.5 hours).  
Ref: <http://docs.fedoraproject.org/en-US/Fedora/14/html/Software_Management_Guide/Configuraci%C3%B3n_de_YUM.html>

# Copying from Jenkins to repo server

<http://schneide.wordpress.com/2013/03/18/1-click-deployment-of-rpms-with-jenkins/>

Another approach. Also includes a ref to RPM project  
<http://forums.jfrog.org/Uploading-RPM-via-Jenkins-Plugin-in-FreeStyle-jobs-td7178919.html>

# YUM commands

Ref: <http://linuxmoz.com/linux-yum-tutorial/> (like this one)  
<http://www.linuxsysadmintutorials.com/how-to-download-rpms-without-installing-them-using-yum/>

You may want to download RPMs only and not install them using yum. Perhaps you’re within a change freeze but would like to ‘warm’ up your yum cache to make your production security upgrades alot faster and smoother.

|  |
| --- |
| The yum-plugin-downloadonly yum plugin provides yum with a –downloadonly option. $ yum install yum-plugin-downloadonly  $ yum install --downloadonly puppet |

|  |
| --- |
| You can also download the RPMs to a directory of your choosing and not /var/cache/yum. $ yum install --downloadonly --downloaddir=/var/tmp/puppet puppet |

|  |
| --- |
| Install local package $ yum localinstall packageName  For any command, packageName can be:  name.arch  name-ver  name-ver-rel  name-ver-rel.arch  name-epoch:ver-rel.arch  epoch:name-ver-rel.arch |

$ yum repolist

Removing a package with an error in preun script  
Instructions are described here:  
<http://serverfault.com/questions/613256/trying-to-remove-packages-with-yum-however-it-is-failing>   
Option “edit 1” worked for me.

# Installing unsiged RPM’s

<https://ask.fedoraproject.org/en/question/31116/how-do-i-install-unsigned-packages-using-yum/>

# RPM Groups

Creating a yum repository (repo) and Creating a yum group to install + kickstart <http://ramblings.narrabilis.com/creating-a-yum-repository-repo-and-creating-a-yum-group-to-install-kickstart>

# RPM Commands

List package info (e.g. app1-0.1-0.x86\_64)  
$ rpm -qp *filename.rpm*

List scriplets  
$ rpm -qp --scripts *filename.rpm*

List content  
$ sudo rpm -qp --list *filename.rpm*

List requires libraries  
$ sudo rpm -qp --requires *filename.rpm*

List provided libraries *$ rpm -qp --provides filename.rpm*

# Private Libraries

Application requires private libraries. Both packaged in same RPM. But these libraries must not be exposed (provided) as there are not public.  
<http://fedoraproject.org/wiki/Packaging:AutoProvidesAndRequiresFiltering>

**%global** *\_privatelibs libnm-iodine-properties[.]so.\**

**%global** *\_\_provides\_exclude ^(%{\_privatelibs})$*

**%global** *\_\_requires\_exclude ^(%{\_privatelibs})$  
  
h t t ps: / /apps.fedoraproject.org/packages/NetworkManager-iodine/sources/spec*

# Commands used in RPM scripts

<http://linuxcommand.org/man_pages/chkconfig8.html>