Table of Contents

Change History

2009/07/25 – Created the FC11 version and published at fedoraforum.org
2010/11/24 – Updated for FC14 for release here

Document Source

At the time of publication this document was available at
http://www.mdickinson.dnsalias.org/public/doc/books/SettingUpALocalRepo.pdf

Whats Covered in here – Purpose of document

First off this is not another document on how to replicate a local mirror of the Fedora Updates repository. I don't have the bandwidth for that.

It may not even be good practise, but it has quartered my download usage so I use it.

This document was created for my personal needs; which is that I have Four Fedora servers and did not want to have to pull down gigabytes of updates to all four when packages were updated, only to one that would then provide them to the others. The document covers

- how to use only one server to store all the files downloaded to it by a 'yum update' and automatically repackage them into a local repository (not the entire updates repository, and not just from Fedora, this covers all rpms from all repositories that have been upgraded by yum)
- how to get all your other servers to pull their yum updates from the local repository on your home network instead of chewing up ISP bandwidth allowances

So basically four servers can use yum update with only one set of files being downloaded, and all those updates are stored locally for if I need to rebuild a (or install a new) server.

By following this document the following benefits can be obtained

- you generally only need to do a 'yum update' from the internet to one server
- using 'yum update' on any other servers will use your local server for the packages needed for any updates
- on the odd occasion a pre-requisite is needed by one of your other servers for a package in your local repository this method still allows yum to satisfy those dependencies from the internet.

Especially beneficial for those damn huge OpenOffice updates.

On the server that will be retrieving yum updates

Install lots of software

Yum will only update the software that is installed. You need to make sure that the server you will be running the internet facing 'yum update' on has installed upon it all the packages that are likely to exist on all the other servers you intend to update from this local repository.

You also must have the createrepo utility. If you don't have /usr/bin/createrepo then 'yum install createrepo' before proceeding (or you cannot create a repo so why bother reading this).

Change yum to not delete cached rpm files

You really need to do this as soon as you have installed the Fedora operating system on the server you intend to download the updates to, to ensure that all updates and dependencies are captured.
I chose to use my main desktop server for this, immediately after it was upgraded to FC11 and no updates had been installed yet.

> **vi /etc/yum.conf**
> change **keepcache=0 to keepcache=1**

With that change all RPMs pulled down will be in directories under the /var/cache/yum directory and you can use those to create a local repo to update your other servers from.

Note1: while purists would say that ideally you would do this on a server, not a desktop; servers normally (certainly mine do) have the minimum number of packages installed to do their function, wheras the desktop I installed virtually eveything (ie: X, OpenOffice etc a server doesn't need) to ensure I got updates for them; you cannot really use a server.

Note2: I also set the keepcache=1 on all my servers so I could capture RPMs downloaded from the internet on those servers also, so I could merge them with my main local RPM updates collection for use by the other servers.

Configure the web server to server your relocal repository

This of course needs to be done on the server you are pulling the yum updates down to. Do NOT set it up on the machine that will be serving out this repository.
I am assuming you will be using the default apache2 directory structure in /var/www.

mkdir /var/www/html/tftpboot/FC14-updates
mkdir /var/www/html/tftpboot/FC14-updates/Packages
chkconfig httpd on
service start httpd

Open port 80 to allow incoming requests

By default (on FC11 that this document was origionally written for anyway) the Fedora firewall blocks port 80, so yum updates from all other local servers failed. So use 'system administration/firewall' to open the HTTP port so yum can actually go and get updates.

If you have an better firewall configuration tool of course just allow incoming http requests from your home subnet.

Automate moving RPMs from cache to your local repository

To do that a simple script to locate the new rpms is all that is needed.
However as I have been using this method for quite a few years now my script has evolved to do pretty much everything for managing the local repo. So as I am updating this document for FC14 I have replaced the origional tiny sample script with the one I use myself here.
The script can be run on demand as needed or from cron if you do daily yum updates.

```
#!/bin/bash
# ************************************************************************
#
# maintain_local_repo - Version 1.0.0 - Mark Dickinson, 2009
#
# Basically the rebuild_local_repo script will always merge new updates into
# the local repository. This can (and does) result in multiple versions
# of the same package being stored in the local repository. The only
# impact of that is really disk space usage; which for something like
# openoffice/eclipse/java can be large.
#
# This script will clean the obsolete RPM files from the local repo
# directory by checking every RPM to see if it is the currently
# installed RPM; and queueing it in a work file for deletion if it
# is not currently installed.
# Thats the checkrepo opton. It builds the work file.
# cleanrepo uses that work file to delete the obsolete files (after a backup)
# restorerepo is there for if you deleted something you needed.
#
# And so it seemed silly to have the seperate rebuild_local_repo script when I
# had this one, so I have merged that into here also, with updaterepo option.
#
# checkrepo - produce a list of files the script deems obsolete
# cleanrepo - backup all fies, then deleet all in the obsolete list
# restorerepo YYYYMMDD - oops, restore all RPMs in a dtatestamped backup from cleanrepo
# backuprepo - just take a backup of current RPMs to a datestamped file
# updaterepo - the origional rebuild_local_repo script merged in, will copy all
#              the new packages from /var/cache/yum into our web server local_repo
#              and rebuild the repo .xml file so they can be used by my other servers
#
# CAUTION: using cleanrepo may break dependencies needed by the local repo
#          from you other servers. I don't use it anymore (well not often,
#          but it does a backuprepo first and restorerepo does work :-)
# ************************************************************************
command="$1"     # checkrepo, cleanrepo or restorerepo
option="$2"      # only used by restorerepo, the date to restore from

# ------------------------------------------------------------------------
# Global variables needed, CUSTOMISE FOR YOUR ENVIRONMENT
# ------------------------------------------------------------------------

# --- We must have a workfile to store obsolete rpm names in temporarily
WORKFILE="/tmp/rpmcheck.data"        # our jolly work file, can be anywhere
# --- The LOCAL_REPO is the root directory of your local repository,
#     which obviously is where in the web server directory structure you
#     chose to put it, this is the directory immediately above the packages
#     directory the rpms will be stored in
releasever=`cat /etc/fedora-release | awk {'print $3'}`
```

```
LOCAL_REPO="/var/www/html/tftpboot/FC${releasever}-updates"  # the root of out local
repository
# --- You should not need to change this, this should be standard on
#     all FCnn systems
#YUM_CACHE_DIR="/var/cache/yum"       # where yum cache is, with yums keep cache option...
YUM_CACHE_DIR="/var/cache/yum/i386/${releasever}"   # UPDATED FOR FC14
                                     # ...all the rpm files updated/installed are here
# --- DO NOT CHANGE THIS. If you setup your LOCAL_REPO cariable correctly this is correct
REPOSDIR="${LOCAL_REPO}/Packages"    # and the packages directory is under that

# ------------------------------------------------------------------------
# The sanity check section.
# ------------------------------------------------------------------------
if [ ! -d ${REPOSDIR} ];
then
   echo "You need to create the directory structure !"
   echo "There is no ${REPOSDIR} directory."
   exit 1
fi

# ------------------------------------------------------------------------
# And then do what we were asked to do.
# ------------------------------------------------------------------------
case "${command}" in
   "backuprepo")
        datenow=`date +"%Y%m%d"`
        cd ${LOCAL_REPO}
        tar -zcf packages_${datenow}.tar.gz Packages/*.rpm
        if [ $? -ne 0 ];
        then
           echo "**ERROR** unable to take a backup of existing RPM files"
           echo "Review previous errors"
           exit 1
        fi
        echo "Backup is stored in ${LOCAL_REPO}/packages_${datenow}.tar.gz"
        exit 0
        ;;
   "checkrepo")
        # (1) Check all RPM files in the REPOS directory to see
        #     if they match the installed version, if not then
        #     assume they have been superceeded and write them
        #     to the work file.
        echo "*** Checking for obsoletet packages in the local repo ***"
        if [ -f ${WORKFILE} ];
        then
           /bin/rm ${WORKFILE}
        fi
        cd ${REPOSDIR}
        ls *rpm | sed -e 's/\.rpm//' | while read rpmname
        do
           sink=`rpm -q ${rpmname}`
           if [ $? -ne 0 ];
           then
              echo "${rpmname}.rpm" >> ${WORKFILE}
              echo "${rpmname} is no longer on this server, pending clean"
           fi
        done
        if [ -f ${WORKFILE} ];
        then
           counter=`cat ${WORKFILE} | wc -l`
           echo "There are ${counter} obsolete RPM files found"
```

```
         echo "Review ${WORKFILE} before running $0 cleanrepo"
      else
         echo "No obsolete packages were found for removal, you are all good."
      fi
      ;;
"cleanrepo")
      if [ -f ${WORKFILE} ];
      then
         echo "*** Backing up current RPM files before being destructive ***"
         $0 backuprepo
         if [ $? -ne 0 ];
         then
            # error messages will have already been written
            exit 1
         fi
         echo "*** Removing obsolete RPM files from the local repo"
         cat ${WORKFILE} | while read filename
         do
            if [ -f ${REPOSDIR}/${filename} ];
            then
               /bin/rm ${REPOSDIR}/${filename}
               echo "Removed ${filename} from local repo"
            else
               echo "*** USER LAZYNESS ERROR ***"
               echo "File ${filename} no longer exists, therefore work file out of date !"
               echo "BACKOUT STARTED"
               datenow=`date +"%Y%m%d"`   # oops, its only known to backout proc
               $0 restorerepo ${datenow}
               echo "BACKOUT COMPLETED"
               echo "Use $0 checkrepo to create an up to date work file"
               exit 1
            fi
         done
         # don't need the work file, do not leave it lying around
         /bin/rm ${WORKFILE}
         echo "*** Rebuiling local repo with the remaining files ***"
         createrepo ${LOCAL_REPO}
      else
         echo "No obsolete RPM filelist for the local repo, nothing to do."
         echo "Did you run $0 checkrepo first ?"
      fi
      ;;
"restorerepo")
      if [ "${option}." == "." ];
      then
         echo "**ERROR** I need the date to restore from !"
         echo "  Syntax: $0 restorerepo YYYYMMDD"
         echo "  Available files are..."
         ls ${LOCAL_REPO}/packages*tar.gz
         exit 1
      fi
      cd ${LOCAL_REPO}
      if [ -f packages_${option}.tar.gz ];
      then
         echo "Beginning restore from packages_${option}.tar.gz"
         tar -zxf packages_${option}.tar.gz
         echo "Rebuiling the local repo to include the restored packages"
         createrepo ${LOCAL_REPO}
      else
         echo "There is no backup file packages_${option}.tar.gz"
         echo "Restore not possible !"
```

```
           exit 1
        fi
        ;;
   "updaterepo")
        # Do nothing if yum is running, there may be incomplete packages
        # it is still retrieving.
        yumactive=`ps -ef | grep "yum" | grep -v "grep"`
        if [ "${yumactive}." != "." ];
        then
           echo "****** yum still running ******"
           echo "${yumactive}"
           echo "So not migrating new files at this time."
           exit 1
        fi
        #
        # Merge all RPM files in yum cache with the local repo copies
        find ${YUM_CACHE_DIR} -name "*.rpm" -type f | while read fname
        do
           fbase=`basename ${fname}`
           cp -p ${fname} ${REPOSDIR}
           isok=$?
           if [ ${isok} -eq 0 ];
           then
              chcon system_u:object_r:httpd_sys_content_t:s0 ${REPOSDIR}/${fbase}
              chown apache:apache ${REPOSDIR}/${fbase}
           fi
        done
        # Delete all the caches stuff from yum cache
# 2010/11/14 Don't clear anything, just copy only from above
#        yum clean dbcache
#        yum clean all
        #
        # Create new repo details for it
        # First check for any obsoletes.
###        $0 checkrepo
        # If there were no obsoletes found by checkrepo it will not do
        # a cleanup and run createrepo, so if no obsoletes were found
        # we run createrepo to create the new repo .xml file with the
        # new packages we just copied into it.
        if [ -f ${WORKFILE} ];
        then
           # remove obsoletes and rebuild the repo .xml file
           $0 cleanrepo
        else
           # just create the new .xml file with the new packages
           echo "Creating new repo file"
           createrepo ${LOCAL_REPO}
        fi
        ;;
   *) echo "************************************************************"
        echo "* DANGEROUS UTILITY: ONLY RUN IF YOU KNOW WHAT YOU ARE DOING *"
        echo "************************************************************"
        echo "$0 checkrepo"
        echo "   Scans all files in the local repo to see if they are"
        echo "   installed on this server, if not they will be queued"
        echo "   to a work file for review, and with cleanrepo deletion"
        echo "$0 cleanrepo"
        echo "   uses the obsolete RPM list created from the checkrepo"
        echo "   option to delete all obsolete RPM files from the local"
        echo "   repository, and reruns the createrepo command to resync"
        echo "   the RPM files remaining with the repo .xml contents list"
```

```
        echo "$0 restorerepo YYYYMMDD"
        echo "   Ok, you deleted something important. This will restore"
        echo "   all the RPM files backed up from the local repo as of the"
        echo "   date the cleanrepo was run to backup and clean up old RPMs"
        echo "   It will rerun createrepo to merge the restored files into"
        echo "   the .xml repo list with whatever files exist in the local"
        echo "   repo (it will not delete what you have recently pulled down)"
        echo "$0 backuprepo"
        echo "   Create a timestamped backup of the local repo."
        echo "   NOTE: done automatically if the cleanrepo command is used"
        echo "$0 updaterepo"
        echo "   Moves all the RPM files from /var/cache/yum to the local"
        echo "   repository, removed obsolete (replaced) packages from the"
        echo "   local repository, and rebuilds the .xml package list."
        echo "   After which all your other servers that use the local repository"
        echo "   will get the updates for those packages from there instead of"
        echo "   using your internet bandwidth."
        exit 1
        ;;
esac
exit 0
```

Now that you have read through the entire script, it is available for download at if you need it from my (intermittently online) server at
http://www.mdickinson.dnsalias.org/public/doc/books/maintain_local_repo.txt

Thats almost it for the update server

After you do your first yum update on your update server, and copy all the RPM files into your new local Packages directory, and run createrepo (or use the above script to do the hard work); then your web server has a RPM repo that can be used by your other servers.

So when done carry on through this document to setup the servers that will update from this one.

Configuring the other servers to use the local repository

Install yum-plugin-priorities

On each of the servers you intend to update from your new local repository you must, if it is not already installed

yum install yum-plugin-priorities

This allows you to prioritise your yum repo files so you can force the local repository to be searched first.

You can check if it is already installed with 'rpm -qa | grep “^yum”'

Notes: in FC13 and earlier this package was yum-priorities. Function seems to be the same.

Create /etc/yum.repos.d/local-updates.repo

The contents of this should point the baseurl to the server you will be serving your local rpm files from...
for example the contents of mine is below. With the priority set to 50 means it will be searched prior to any of the internet ones.
Note: I have found I need the version=14 value.

```
[local-updates]
name=LOCAL RPM Update Packages  -
version=14
failovermethod=priority
baseurl=http://192.168.1.186/tftpboot/FC$releasever-updates/
enabled=1
gpgcheck=0
retries=1
priority=50
```

Now yum will download rpms from your local web server that you setup earlier before using any internet connected ones.

Test it, make sure it uses your local repositories

After you have done all the hard work above, go to one of your other servers that you have setup the local updates file in /etc/yum.repos.d and play with yum. You should find most updates are now served from your local repository instead of the internet.

```
[root@falcon books]# yum check-update
Loaded plugins: refresh-packagekit

ConsoleKit.i686                                0.4.2-3.fc14                   local-updates
ConsoleKit-libs.i686                           0.4.2-3.fc14                   local-updates
ConsoleKit-x11.i686                            0.4.2-3.fc14                   local-updates
abrt.i686                                      1.1.14-1.fc14                  local-updates
abrt-addon-ccpp.i686                           1.1.14-1.fc14                  local-updates
abrt-addon-kerneloops.i686                     1.1.14-1.fc14                  local-updates
abrt-addon-python.i686                         1.1.14-1.fc14                  local-updates
abrt-desktop.i686                              1.1.14-1.fc14                  local-updates
abrt-gui.i686                                  1.1.14-1.fc14                  local-updates
abrt-libs.i686                                 1.1.14-1.fc14                  local-updates
abrt-plugin-bugzilla.i686                      1.1.14-1.fc14                  local-updates
abrt-plugin-logger.i686                        1.1.14-1.fc14                  local-updates
abrt-plugin-runapp.i686                        1.1.14-1.fc14                  local-updates
atk.i686                                       1.32.0-1.fc14                  local-updates
atk-devel.i686                                 1.32.0-1.fc14                  local-updates
checkstyle.noarch                              5.1-2.fc14                     updates
clutter-gtk.i686                               0.10.8-2.fc14                  local-updates
evolution.i686                                 2.32.1-1.fc14                  local-updates
evolution-data-server.i686                     2.32.1-1.fc14                  local-updates
evolution-data-server-devel.i686               2.32.1-1.fc14                  local-updates
evolution-data-server-doc.noarch               2.32.1-1.fc14                  local-updates
evolution-help.noarch                          2.32.1-1.fc14                  local-updates
evolution-perl.i686                            2.32.1-1.fc14                  local-updates
fprintd.i686                                   0.2.0-2.fc14                   local-updates
gdb.i686                                       7.2-26.fc14                    local-updates
geoclue.i686                                   0.12.0-3.fc14                  local-updates
geoclue-devel.i686                             0.12.0-3.fc14                  local-updates
gnome-applets.i686                             1:2.32.0-2.fc14                local-updates
gnome-dvb-daemon.i686                          0.1.21-1.fc14                  local-updates
gssdp.i686                                     0.8.1-1.fc14                   local-updates
gssdp-devel.i686                               0.8.1-1.fc14                   local-updates
gtk-vnc.i686                                   0.4.2-1.fc14                   local-updates
gtk-vnc-python.i686                            0.4.2-1.fc14                   local-updates
gtkhtml3.i686                                  3.32.1-1.fc14                  local-updates
gvfs.i686                                      1.6.6-1.fc14                   local-updates
...etc...
```

What you should expect to see, as in the above screen cut/paste is that most of your other servers should now update from the local repository instead of the internet, but yum will still satisfy dependencies as needed from the internet. In the case above one (OK, if I pasted the entire list there were two), but you can see the advantage of serving updates from your local repository already... and if this was server2 being updated if you copies the two extra RPMs from the /var/cache/yum... and applied them to your local repo no other servers would need to go out to the internet.

HOWEVER all kernel updates are always served from the Fedora updates repository; I guess thats a wisely installed safeguard. Just so you know, expect that.

Other things you may want to do

As I noted at the start of this document I also turn the keep cache option on for all of my servers, this also is for my use as I use network boot to trigger a network install/kickstart my web server from a standard base FC14 repository, and want all the updates available locally rather than having to download all the packages again after each monthly test of my rebuild procedures.

So if you follow that path you will need scheduled jobs on all your servers to collect the internet downloaded RPMs from all your other servers and move them onto your local repository server for the next time they are needed.

But as this is not a document on backup strategies I don't intend to cover all your options here.

Possible Problems

The other servers are not using the local repository

If an any time before setting up your local /etc/yum.repo.d/local-updates.repo you has used yum for anything it will have already created a working environment that knows nothing about your local repository. That is easily fixed

yum clean dbcache
yum clean all

and try again.

My main update server no longer updates from the internet

If you have created a /etc/yum.repos.d/local-updates.repo on that main yum update server get rid of it !. You only want that on the servers that will be referencing you local update server, you do not want it on the internet facing update server itself. That was mentioned earlier !.