Red Hat Enterprise Linux Rebuild mini-HOWTO

Michael Redinger

Michael.Redinger@uibk.ac.at

This document describes how to build a Linux server based on the SRPMS from Red Hat Enterprise Linux 2.1.

Table of Contents

1. Introduction

Red Hat changed their support cycle for Red Hat Linux in 2002. According to the new policy, updates are only available for 1 year. Also, with the introduction of Red Hat Linux 9, the very nature of their “retail version” changed - it is now much more experimental, not suited for servers anymore.

Red Hat also produces a product that focuses on server systems: Red Hat Enterprise Linux (RHEL) (ES and AS). This can be considered a “real” server system. The update cycles are slower and more conservative, each release is supported for 5 years. That is exactly what a serious server administrator wants. However, these are high priced. As of July 2003, RHEL ES costs $349, RHEL AS a princely $1490 - per server per year.

This might be an acceptable price for one server. But if you have 10+ servers, this is IMO too expensive.

But Red Hat is still very committed to Open Source software and therefore distributes source RPMS for all components of RHEL - including the installer. Because of this it is possible to rebuild a server system based on the RHEL SRPMS.

This document describes one way how this can be done.

Let me repeat that this is about “rebuilding a Linux server based on the Red Hat Enterprise Linux sources”. That means that the resulting system is *not Red Hat Enterprise Linux*. Red Hat clearly states that others must not use the Red Hat trademark for derived work. See their Trademark Guidelines and Policies (http://www.redhat.com/about/corporate/trademark/).

Please note that this document is work in progress and must be considered “alpha”.

1.1. Copyright Information

1.2. Disclaimer

No liability for the contents of this documents can be accepted. Use the concepts, examples and other content at your own risk. As this is a new edition of this document, there may be errors and inaccuracies, that may of course be damaging to your system. Proceed with caution, and although system damage is highly unlikely, the author(s) do not take any responsibility for that.

All copyrights are held by their respective owners, unless specifically noted otherwise. Use of a term in this document should not be regarded as affecting the validity of any trademark or service mark.

Naming of particular products or brands should not be seen as endorsements.

1.3. New Versions

The newest version of this HOWTO will be made available on the website of the University of Innsbruck, in a variety of formats:

- HTML (http://www.uibk.ac.at/zid/software/unix/linux/rhel-rebuild.html).
- PDF (http://www.uibk.ac.at/zid/software/unix/linux/rhel-rebuild.pdf).
- Postscript (A4) (http://www.uibk.ac.at/zid/software/unix/linux/rhel-rebuild.ps).
- SGML source (http://www.uibk.ac.at/zid/software/unix/linux/rhel-rebuild.sgml).

1.4. Credits

Special thanks go to Josep M. (<`websurfer AT navegants DOT com`>) who sent his description on how to use RHL 7.2 as a build host. I actually did never try this (but the mistakes in this description are, of course, mine).

I would like to thank David L. Parsley for various invaluable hints.

Credits are also due to Red Hat and its really excellent employees for building an “open” distribution. The project described herein would not have been possible otherwise.

1.5. Feedback

Feedback is most certainly welcome for this document. Please send your additions, comments and criticisms to the following email address : <`Michael.Redinger@uibk.ac.at`>.

2. Getting Started

2.1. Prerequisites

I assume that you are an experienced system administrator. You should also know how to handle RPM files - this includes installation and (re)building. Otherwise read the *RPM HOWTO* first.

I strongly recommend that you also read the *Burning a RedHat CD HOWTO* first. It describes how to build your own “Red Hat” CDs.

Building a server based on RHEL SRPMS is just a special case of what is described in *Burning a RedHat CD HOWTO*. Therefore I will not repeat the valuable information given in that document. Instead I will only focus on information that is not available therein.

You will also need a build server - see below for details.

2.2. Build Server

You will need a server for building the SRPMS and generate the distribution. I am using a RHEL AS system for this task. This means that you have to buy one RHEL system. However, I thought that this is definitely worth it: I can be sure that my build is very close or almost identical to the “real” product.

As RHEL 2.1 is actually based on RHL 7.2, it is also possible to use RHL 7.2 as a build host.

You should do an “**Everything**” installation to include all packages (especially libraries and `*-devel`) that are needed for rebuilding the SRPMS.

If you want to share the created RPMS with other you should include all available languages during installation.

I also highly recommend applying all available RHEL updates. First I thought it should be better to rebuild the SRPMS on a “plain” RHES installation, but I found that this does not work very well.

2.3. Fix Red Hat Bugs

First of all, let me repeat that you have to apply all updates available. Otherwise many packages will not build correctly.

There is a link missing for the **gdk-pixbuf** include files. If you do not add this link, most GNOME packages will not build:

```
( cd /usr/include/ ; ln -s gdk-pixbuf-1.0/gdk-pixbuf )
```

3. Building the RPMS

3.1. Downloading the SRPMS

First of all, download the SRPMS for RHEL AS from ftp://ftp.redhat.com/pub/redhat/linux/enterprise/ or any mirror. Also download the updated SRPMS from ftp://ftp.redhat.com/pub/redhat/linux/updates/enterprise/.

3.2. Create your build tree

You probably only want to build the latest version of a package - so merge the original SRPMS directory and the updated SRPMS and remove the old SRPMS.

I used **updateCD.sh** from the **rhcd-scripts** package mentioned in the *Burning a RedHat CD HOWTO*. You can get it from the LDP CVS archive (http://cvsview.tldp.org/index.cgi/*checkout*/LDP/howto/docbook/RedHat-CD-HOWTO/rhcd-scripts.tar.gz?rev=1.1&sortby=author&content-type=application/x-tar).

Unpack **rhcd-scripts.tar.gz** somewhere on the build server (for example `/usr/local/`. Now change to `rhcd-scripts/rpmvc/` and run **make** to build the program **rvc**.

Then create the configuration file `/usr/local/rhcd-scripts/rhcd.conf`. If the scripts are installed in `/usr/local/rhcpd-scripts/`, the original source RPMS are in `/usr/local/es/2.1AS/i386/SRPMS/` and the source RPMS for the updates in `/usr/local/updates/SRPMS/`, then the configuration file looks like this:

```
UPDDIR=/usr/local/updates/
ARCH=SRPMS
RHROOT=/usr/local/es/2.1AS/i386/SRPMS/
OLDDIR=/usr/local/SRPMS.old
CHECKSIG=0
USEGPG=0
RVC=/usr/local/rhcd-scripts/rpmvc/rvc
```

You also have to create the directory `/usr/local/SRPMS.old/`. This is the directory the old SRPMS are move to.

Now change to your updates SRPMS directory and remove the files `kernel-2.4.18-e.31.src.rpm` and `mkinitrd-3.4.24-0.ia64.2.src.rpm`. These SRPMS are for the ia64 platform only.

There is a bug in the **updateCD.sh**. Almost at the end of the file there is a comment "*Copy new boot image files to the right place* ...". The next line is wrong and has to be replaced. Here is my patch:

```
--- updateCD.sh.orig    2003-07-24 14:35:25.000000000 +0200
+++ updateCD.sh 2003-07-24 15:20:23.000000000 +0200
@@ -139,7 +139,7 @@
 fi

 # Copy new boot image files to the right place... (only for 6.x)
-if [ -d $IMGDIR ] ; then
+if [ -n "${IMGDIR}" -a -d "${IMGDIR}" ] ; then
     # directory where the updates have to be copied to
     ANACONDIR=${RHROOT}/misc/src/anaconda
     for bootimage in ‘ls -rt ${IMGDIR}/*‘ ; do
```

Now you can merge the two directories. Change to your **rhcd-scripts** directory, set the variable for the configuration file and run **updateCD.sh**:

```
cd /usr/local/rhcd-scripts/
export RHCDPATH=/usr/local/rhcd-scripts
./updateCD.sh
```

This worked well for most of the packages. But I found that for some packages the new package was moved to the SRPM directory but the old one was not removed. So you might want to change to your SRPM directory and run the following script to see if you have to move any packages manually:

```
OLDNAME=""
NAME=""
for file in *.rpm ; do
   OLDNAME=$NAME
   NAME=$(rpm -pq --queryformat ’%{NAME}’ ${file})
   if [ "${NAME}" = "${OLDNAME}" ] ; then
      echo "${NAME}"
   fi
done
```

3.3. Rebuilding the SRPMS

Building the RPMS from the SRPMS is actually very easy - simply use **rpmbuild --rebuild [file]** for all the files:

```
REPORTS=/usr/local/build.reports
[[ -d ${REPORTS} ]] || mkdir ${REPORTS}
for srpm in *.src.rpm ; do
  rpmbuild --rebuild ${srpm} > ${REPORTS}/${srpm}.out 2>&1
done
```

> **Note:** It is highly recommended that you do *not* build the packages as root but use a different user (and choose a **%_topdir** in your `~/.rpmmacros`).

You probably want to move the SRPMS that built successfully to another directory. Suppose the SRPMS are in `/usr/local/es/2.1AS/i386/SRPMS/`. Then make a directory `/usr/local/es/2.1AS/i386/SRPMS.done/` and start the build:

```
REPORTS=/usr/local/build.reports
[[ -d ${REPORTS} ]] || mkdir ${REPORTS}
DONE=/usr/local/src/SRPMS.done
for srpm in *.src.rpm ; do
  rpmbuild --rebuild ${srpm} > ${REPORTS}/${srpm}.out 2>&1
  if [ "${?}" = "0" ] ; then
    mv ${srpm} ${DONE}
  fi
done
```

You will find the successfully built RPM packages in `/usr/src/redhat/RPMS/i386/` (for “i386”) and `/usr/src/redhat/RPMS/noarch/` (for architecture-independant packages).

3.3.1. RHEL 2.1AS build host

tetex probably will not build correctly. Install the **tetex** SRPM:

```
rpm -ivh tetex-*.src.rpm
```

Then edit `/usr/src/redhat/SPECS/tetex.spec`. There you should find the following line:

```
%define enable_japanese 1
```

Replace “**1**” with “**0**”. This should do the trick, so you can now build the RPM:

```
rpmbuild -ba /usr/src/redhat/SPECS/tetex.spec
```

This disables Japanese support for TeX. As I do not need this, that is the easiest workaround. If anybody finds a better solution, please let me know.

When using RHEL AS as your build server, the packages **ethtool** and **redhat-config-network** will not build either. You have to downgrade your **kernel-headers** to the version that was shipped with RHEL AS.

There is also a problem building the updated **im** package on RHEL AS. Install the **im** SRPM:

```
rpm -ivh im-*.src.rpm
```

Edit the the **im.spec** SPEC file and find the line that starts with **BuildRequires** and replace **glibc-kernheaders** with **kernel-headers**. Now the package should build correctly:

```
rpmbuild -ba /usr/src/redhat/SPECS/im.spec
```

RHEL includes some packages that are optimized for Pentium III systems - "i686" RPMS:

- glibc
- kernel
- openssl

To build the "i686" RPMS, use the following command:

```
rpmbuild --target i686 [file]
```

Now all RPMS except **compat-glibc**, **compat-libs** and **nautilus** should have been built successfully. I just took these RPMS from RHL 7.2 (the *compat* RPMS are actually the same for RHL 7.2 and RHEL).

Now there should be only the **rpmdb** left to be rebuilt. See below for a description how to rebuild it.

3.3.2. RHL 7.2 build host

Section is based on what Josep M. told me.

He started with a "plain" 7.2 installation. He also installed **dietlibc**, **modutils-devel** and **libmad** (he recompiled the **dietlibc**, **modutils** and **libmad** packages from RHEL). Note that you have to rebuild and install **dietlib** before rebuilding **modutils**.

Some packages did not compile correctly. So he now applied all updates available for RHL 7.2.

Josep also updated to the **libxml2** and **libxml2-devel** from RHEL 2.1.

Now he started building the SRPMS again.

Apache did not build correctly, he ended up using the version **1.3.23** (a previous update version).

There were also problems building Mozilla. He finally compiled and installed the following packages from the RHEL 2.1 SRPMS: **mozilla-nspr**, **mozilla-devel**, **mozilla**, **mozilla-psm**, **mozilla-mail**, **mozilla-nss**, **mozilla-chat**, **kdebindings-kmozilla**, **nautilus-mozilla**, **galeon**, **nautilus** and **nautilus-devel**.

As described above, you also have to build the "i686" RPM packages.

The **compat-glibc** and **compat-libs** cannot be built on RHL 7.2 either, just use the binary packages included in RHL 7.2.

.

See below for how to build the **rpmdb** package.

4. Build the distribution

4.1. Build the distribution directory

Let us suppose you want to use the directory `/usr/local/es/2.1AS/i386/` for your newly built distribution. Then you would have to put the RPMS in `/usr/local/es/2.1AS/i386/RedHat/RPMS/` (there is a reason for adding "i386" here - it is used by the **rpmdb** SPEC file). Simply copy all the RPMS to that directory (hint: do not forget the **noarch** and **i686** packages) and set the permissions to **644**.

Now you can also build your **rpmdb** package. Install the SRPM first:

```
rpm -ivh rpmdb-redhat*.src.rpm
```

Now rename the SPEC file:

```
SPECDIR="/usr/src/redhat/SPECS"
mv ${SPECDIR}/rpmdb-redhat.spec.in ${SPECDIR}/rpmdb-redhat.spec
```

Then edit the SPEC file and add the following lines at the beginning (they should be already there, but they are commented out):

```
%define rpmdbname               redhat
%define rpmdbversion            2.1AS
%define rpmdbtree               /usr/local/es/2.1AS
```

Now you can build the **rpmdb**:

```
rpmbuild -ba /usr/src/redhat/SPECS/rpmdb-redhat.spec
```

4.2. Build the installer

When I started working on "rhel-rebuild", I thought this would be the biggest problem. As you know, you must not reuse any part of RHEL but the SRPMS. So I thought it would be necessary to adapt the Red Hat Linux 7.3 installer.

But there is an easy solution for this - Red Hat Enterprise Linux includes a script that generates the installer - including the floppy disk images. Make sure you have **anaconda-runtime** installed. Then run the following programs:

```
/usr/lib/anaconda-runtime/buildinstall /usr/local/es/2.1AS/i386/
/usr/lib/anaconda-runtime/genhdlist /usr/local/es/2.1AS/i386/
```

4.3. Add the comps file

Now you are still missing the file `RedHat/base/comps` in your distribution directory. Again, see the *Burning a RedHat CD HOWTO* for a description of what this is all about.

You cannot use the `comps` file that comes with RHEL 2.1. So I decided to use the `comps` from RHL .7.2.

This is actually quite easy - you basically just have to remove the packages that are not included in RHEL 2.1. You also have to add a section "Advanced Server" to keep the installer satisfied.

I also added some packages to the `comps` file that are included in RHEL 2.1 but not in RHL 7.2. Most of these packages are included in RHL 7.3 or RHL 9, so I had a look at these `comps` too. (If I had to do this again I would probably start with a RHL 7.3 comps file.)

You can download my comps file here: http://www2.uibk.ac.at/zid/software/unix/linux/comps-rhel-rebuild-2.1 (comps-rhel-rebuild-2.1). Copy this file to `/usr/local/es/2.1AS/i386/RedHat/base/comps`.

If you change the `comps` file you should check if there are any problems with it. There is a script included in the **anaconda-runtime**, **check-repository.py**, that does exactly this. You have to change one line to make it work. Edit the file `/usr/lib/anaconda-runtime/check-repository.py` and remove the following line:

```
import todo
```

Now run the script:

```
/usr/lib/anaconda-runtime/check-repository.py /usr/local/es/2.1AS/i386/
```

5. Further steps

I always use network based installs, so I am basically done now. But you may want to burn this distribution on CD etc. Please see *Burning a RedHat CD HOWTO* for details.

5.1. Enjoy!

Well, that is it, folks! Build your installation floppy disc (or boot from CD-ROM, use PXE network boot - whatever) and you are (hopefully) done. Enjoy!

6. Problems

When using the latest kernel (**2.4.9-e.25**), `libredhat-kernel.so.1`, which is needed by **glibc** is included in the **kernel-BOOT** package (which introduces a new dependency).

When first booting a newly installed server, **sleep** complains that it cannot find `libredhat-kernel.so.1`. This works fine for subsequent boots.

7. Frequently Asked Questions

> **Note:** FIXME: get some FAQs ...