**Creating A Local and HTTP RedHat Yum Repository**

by [**Rus**](http://www.idimmu.net/author/rus/)[**+**](https://plus.google.com/u/0/111945257118069299216) on October 20, 2009 in [**Linux**](http://www.idimmu.net/category/systems/linux/), [**Redhat**](http://www.idimmu.net/category/systems/redhat/)

We’re rolling out a [RedHat](http://www.redhat.com/) platform for a major product delivery to a client in the next couple of weeks, which is a great chance for me to build a fresh platform using all the skills I’ve acquired and actually do things properly (TM).

We’re using [RedHat Enterprise Linux 5](http://www.redhat.com/rhel/) because this is a very critical deployment and we want an Operating System that is certified to both our hardware and software applications, something that unfortunately [Ubuntu](http://www.ubuntu.com/) does not deliver.

The first stage of the deployment is to create our own RedHat RPM repository on the local network, so we can easily create servers.

I’m going to cover setting up 2 types of repository, initially a file system based repo and then an http based repo so that other servers can talk to the repo and snag RPMs from it.

To start with a repository is **just** a path with RPMs in it, regardless of file hierarchy, so we need to create this. I’m working with the RHEL 5 DVD so my first task is to simply copy the contents of the DVD on to the server’s hard drive.

First we mount the DVD

[rus@redhat ~]$ sudo mkdir /media/cdrom  
[rus@redhat ~]$ sudo mount /dev/cdrom /media/cdrom/

And then we copy across the entire of the DVD to where we want the file system based repo to be!

[rus@redhat ~]$ sudo mkdir -p /var/repo/rhel5  
[rus@redhat ~]$ sudo cp -r /media/cdrom/\* /var/repo/rhel5  
[rus@redhat ~]$ find /var/repo/rhel5 -name \*.rpm | wc -l  
3180

Ok, now we have a lovely directory structure with over 3000 RPMs just gagging to be installed in it ;)

Next up we need to create the Yum repository metadata, to do this we first need to install a tool called[createrepo](http://createrepo.baseurl.org/). Fortunately this is on the DVD image we just copied to our hard drive so we can sneakily install it with *rpm -i* ;)

[rus@redhat Server]$ cd /var/repo/rhel5/Server/  
[rus@redhat Server]$ sudo rpm -i createrepo-0.4.11-3.el5.noarch.rpm  
warning: createrepo-0.4.11-3.el5.noarch.rpm: Header V3 DSA signature: NOKEY, key ID 37017186

Tada, createrepo is now installed! Do not worry about the *NOKEY* error, that will be explained and fixed shortly!

To create the metadata we simply run *createrepo .* in the repo directory.

[rus@redhat Server]$ cd /var/repo/  
[rus@redhat repo]$ sudo createrepo .  
3180/3180 - rhel5/VT/Virtualization-it-IT-5.2-11.noarch.rpm .rpm3.x86\_64.rpm  
Saving Primary metadata  
Saving file lists metadata  
Saving other metadata  
[rus@redhat repo]$ ls  
repodata rhel5  
[rus@redhat repo]$ ls -al repodata/  
total 13460  
drwxr-xr-x 2 root root 4096 Oct 20 23:00 .  
drwxr-xr-x 4 root root 4096 Oct 20 23:00 ..  
-rw-r--r-- 1 root root 3178251 Oct 20 23:00 filelists.xml.gz  
-rw-r--r-- 1 root root 9431119 Oct 20 23:00 other.xml.gz  
-rw-r--r-- 1 root root 1108342 Oct 20 23:00 primary.xml.gz  
-rw-r--r-- 1 root root 951 Oct 20 23:00 repomd.xml

As you can see we have lovely metadata now for our 3180 RPMs!

**repomd.xml** this is the file that describes the other metadata files. It is like an index file to point to the other files. It contains timestamps and checksums for the other files. This lets a client download this one, small file and know if anything else has changed. This also means that cryptographically (ex: gpg) signing this one file can ensure repository integrity.

**primary.xml.gz** this file stores the primary metadata information. This includes information such as name, epoch, version, release, architecture, file size, file location, description, summary, format, checksums header byte-ranges, dependencies, provides, conflicts, obsoletes, suggests, recommends,file lists for the package for CERTAIN files – specifically files matching: /etc\*, \*bin/\*, /usr/lib/sendmail

**filelists.xml.gz** this file stores the complete file and directory listings for the packages. The package is identified by: name, epoch, version, release, architecture and package checksum id.

**other.xml.gz** this file currently only stores the changelog data from packages. However, this file could be used for any other additional metadata that could be useful for clients.

Ok, we now have our file system based repo, so we now need to tell Yum all about it!

Yum repository information is located in */etc/yum.repos.d* as a series of files, it is possible to define multiple repositories in a single file, but I prefer to limit each file to one repository as in my opinion this makes it easier to manage, especially when implementing tools like [Puppet](http://reductivelabs.com/products/puppet).

Our repository is a local rhel repo, so create it *vi /etc/yum.repo.d/rhel-local.repo*

[rhel-local]  
gpgcheck=1  
name=Red Hat Linux $releasever - $basearch - DVD  
baseurl=file:///var/repo

What’s going on here is quite simple:

We’ve created a repo labeled *rhel-local*. We’ve told Yum that the RPMs are signed using GPG, we’ve given it a name, and we’ve defined the baseurl of the repo to be a local file path!

To test we simply run *yum list* and you’ll see that packages are now available in the rhel-local repo!

That’s great, but we have to get on to the real task of creating a HTTP based Yum repository so we can use it to start building servers!! We’re going to serve the repo via the [Apache HTTPD](http://httpd.apache.org/) server so the first logical thing to do is to install it, using our new fancy pants local repo ;)

rus@redhat ~]$ sudo yum install httpd  
Loaded plugins: rhnplugin, security  
This system is not registered with RHN.  
RHN support will be disabled.  
Setting up Install Process  
Resolving Dependencies  
--&gt; Running transaction check  
---&gt; Package httpd.x86\_64 0:2.2.3-31.el5 set to be updated  
--&gt; Processing Dependency: libaprutil-1.so.0()(64bit) for package: httpd  
--&gt; Processing Dependency: libapr-1.so.0()(64bit) for package: httpd  
--&gt; Running transaction check  
---&gt; Package apr.x86\_64 0:1.2.7-11.el5\_3.1 set to be updated  
---&gt; Package apr-util.x86\_64 0:1.2.7-7.el5\_3.2 set to be updated  
--&gt; Processing Dependency: libpq.so.4()(64bit) for package: apr-util  
--&gt; Running transaction check  
---&gt; Package postgresql-libs.x86\_64 0:8.1.11-1.el5\_1.1 set to be updated  
--&gt; Finished Dependency Resolution  
Dependencies Resolved  
=========================================================================  
Package Arch Version Repository Size  
=========================================================================  
Installing:  
httpd x86\_64 2.2.3-31.el5 rhel-local 1.2 M  
Installing for dependencies:  
apr x86\_64 1.2.7-11.el5\_3.1 rhel-local 119 k  
apr-util x86\_64 1.2.7-7.el5\_3.2 rhel-local 75 k  
postgresql-libs x86\_64 8.1.11-1.el5\_1.1 rhel-local 195 k  
Transaction Summary  
=========================================================================  
Install 4 Package(s)  
Update 0 Package(s)  
Remove 0 Package(s)  
Total download size: 1.6 M  
Is this ok [y/N]: y  
Downloading Packages:  
————————————————————————-  
Total 324 MB/s | 1.6 MB 00:00  
warning: rpmts\_HdrFromFdno: Header V3 DSA signature: NOKEY, key ID 37017186  
Public key for apr-util-1.2.7-7.el5\_3.2.x86\_64.rpm is not installed

Oh god no, what is this error? Wait, that looks similar to the **rpm NOKEY** error we saw earlier! Relax, it's simple!

Remember we set the repository to be GPG signed? Well, it is, but we don't have the GPG key installed that we need to use to compare to the signings! This is easily fixable as the GPG keys came on the DVD, which we copied to the hard drive ;)

[rus@redhat ~]$ sudo rpm --import /var/repo/rhel5/RPM-GPG-KEY-redhat-beta /var/repo/rhel5/RPM-GPG-KEY-redhat-release

Now we can retry!

[rus@redhat ~]$ sudo yum install httpd  
Loaded plugins: rhnplugin, security  
This system is not registered with RHN.  
RHN support will be disabled.  
Setting up Install Process  
Resolving Dependencies  
--&gt; Running transaction check  
---&gt; Package httpd.x86\_64 0:2.2.3-31.el5 set to be updated  
--&gt; Processing Dependency: libaprutil-1.so.0()(64bit) for package: httpd  
--&gt; Processing Dependency: libapr-1.so.0()(64bit) for package: httpd  
--&gt; Running transaction check  
---&gt; Package apr.x86\_64 0:1.2.7-11.el5\_3.1 set to be updated  
---&gt; Package apr-util.x86\_64 0:1.2.7-7.el5\_3.2 set to be updated  
--&gt; Processing Dependency: libpq.so.4()(64bit) for package: apr-util  
--&gt; Running transaction check  
---&gt; Package postgresql-libs.x86\_64 0:8.1.11-1.el5\_1.1 set to be updated  
--&gt; Finished Dependency Resolution  
Dependencies Resolved  
==============================================================================  
Package Arch Version Repository Size  
==============================================================================  
Installing:  
httpd x86\_64 2.2.3-31.el5 rhel-local 1.2 M  
Installing for dependencies:  
apr x86\_64 1.2.7-11.el5\_3.1 rhel-local 119 k  
apr-util x86\_64 1.2.7-7.el5\_3.2 rhel-local 75 k  
postgresql-libs x86\_64 8.1.11-1.el5\_1.1 rhel-local 195 k  
Transaction Summary  
==============================================================================  
Install 4 Package(s)  
Update 0 Package(s)  
Remove 0 Package(s)  
Total download size: 1.6 M  
Is this ok [y/N]: y  
Downloading Packages:  
------------------------------------------------------------------------------  
Total 559 MB/s | 1.6 MB 00:00  
Running rpm\_check\_debug  
Running Transaction Test  
Finished Transaction Test  
Transaction Test Succeeded  
Running Transaction  
Installing : apr 1/4  
Installing : postgresql-libs 2/4  
Installing : apr-util 3/4  
Installing : httpd 4/4  
Installed:  
httpd.x86\_64 0:2.2.3-31.el5  
Dependency Installed:  
apr.x86\_64 0:1.2.7-11.el5\_3.1 apr-util.x86\_64 0:1.2.7-7.el5\_3.2  
postgresql-libs.x86\_64 0:8.1.11-1.el5\_1.1  
Complete!

Disco, all done.

RedHat has a quirk though, a default install of most services means that the service will not start if the box is rebooted, in fact after installing an RPM of a service, like Apache it doesn't even start the daemon! So we must do both, register the service to be started each boot and then start the server up!

[rus@redhat ~]$ sudo /sbin/chkconfig httpd on  
[rus@redhat ~]$ sudo /etc/init.d/httpd start  
Starting httpd: httpd: Could not reliably determine the server's fully qualified domain name, using 127.0.0.1 for ServerName  
[ OK ]

Using chkconfig we can tell the system to start the service on boot!

Now we just need to configure Apache to use the repo we've previously set up!

There are an infinite number of ways to configure Apache to serve data, but I'm going to do it the simple and quick way and just use an *Alias* to do it. Create */etc/httpd/conf.d/repo.conf*

Alias /repo/ "/var/repo/"

Now we just need to reload Apache

[rus@redhat ~]$ sudo /etc/init.d/httpd reload  
Reloading httpd: [ OK ]

And we should be able to access the all important *repomd.xml* file with lynx :

[rus@redhat ~]$ lynx http://localhost/repo/repodata/repomd.xml

If Apache throws a *403 forbidden error* error this could very well be [SELinux](http://fedoraproject.org/wiki/SELinux) getting in the way. If you are using SELinux you will need to change the [Security Context](https://fedoraproject.org/wiki/SELinux/SecurityContext) of **all** the repo files so that Apache can read them!

[rus@redhat ~]$ sudo chcon -R -t httpd\_sys\_content\_t /var/repo/

Lynx should now be able to download that file successfully.

The last thing to do now that we can retrieve files is to configure Yum to use the web based repo!

First we're going to disable the existing local file server repo */etc/yum.repos.d/rhel-local.repo* with the*enabled=0* flag:

[rhel-local]  
gpgcheck=1  
enabled=0  
name=Red Hat Linux $releasever - $basearch - DVD  
baseurl=file:///var/repo

And create our new HTTP repo */etc/yum.repos.d/rhel-www.repo*

[rhel-www]  
gpgcheck=1  
name=Red Hat Linux $releasever - $basearch - DVD  
baseurl=http://localhost/repo/

To check it's working we then just issue the following Yum commands to first clean up the repo and to then display their status!

[rus@redhat yum.repos.d]$ sudo yum clean all  
Loaded plugins: rhnplugin, security  
Cleaning up Everything  
[rus@redhat yum.repos.d]$ sudo yum repolist all  
Loaded plugins: rhnplugin, security  
This system is not registered with RHN.  
RHN support will be disabled.  
rhel-www | 951 B 00:00  
rhel-www/primary | 1.1 MB 00:00  
rhel-www 3180/3180  
repo id repo name status  
rhel-debuginfo Red Hat Enterprise Linux 5Server - x86\_64 - Debu disabled  
rhel-local Red Hat Linux 5Server - x86\_64 - DVD disabled  
rhel-www Red Hat Linux 5Server - x86\_64 - DVD enabled: 3,180  
repolist: 3,180

Voila, one HTTP based RedHat Enterprise Linux RPM repository!