RPM and Yum

How to manage your packages without UPS

Nick Urbanik <nicku@nicku.org>

Outline

What is RPM?

- RPM stands for The RPM Package Manager.
  - It used to stand for Red Hat Package Manager
- It is used to manage *packages*.
  - “Oh, I *see*.”
  - . . . no I don’t. . .

What is an RPM Package?

- It's a piece of software, all put in one RPM file
- For example, I am using the... let's see...
  ```
  $ rpm -q evince ↩
  evince-0.5.1-3
  ```
  ...software package to display these notes.
- The *name* of the package is `evince`.
- The *version* of the package is 0.5.1
- The *release* is 3.

An RPM Package usually contains files

- My `evince` package has files inside:

```
$ rpm -ql evince ↩
/etc/gconf/schemas/evince-thumbnailer.schemas
/etc/gconf/schemas/evince.schemas
/usr/bin/evince
...
/usr/share/omf/evince/evince-sr.omf
/usr/share/omf/evince/evince-uk.omf
```

Why should I have packages? Why not just zip files or something?

- We could just have zip files, or tar balls, or something like that, with just the files inside?
- It could work.
- So why have RPM packages?

Advantages of RPM packages

- You can install a new package easily with one operation
- You can uninstall a package just as easily
- the package can contain information about what else it needs to work properly
- RPM won't clobber configuration files, and does sensible things with them
- The RPM system includes a database with lots of information about how the files should be installed, their size, owner, type,. . .

RPM Commands

There are seven main RPM operations:

- For installing software packages:
  - *update*, *install*, *freshen*
- For making *queries*
- for *erasing* packages
- for *verifying* that a software package is correctly installed
- for *checking signatures* — verifying that an RPM package file is really from the people you trust, and not from Joe Trojan Cracker

Queries

- You saw above that I can list the files with
  `$ rpm -ql evince ↩`
- I can also see information about the package with
  `$ rpm -qi evince ↩`
- Try it now!

Install

- This will install a software package, regardless of whether there is a previous version installed
- This is always the right thing to do for kernels.

Update

- Install a package if it is not installed, or
- install a newer version if there is a newer version available. . .
- . . . otherwise, do nothing.
- Let's use this command to install the wonderful Planet Penguin Racer game:
- `$ cd ~/mnt ↩`
- `$ ls -l ↩`
- `$ sudo rpm -Uhv $(ls|grep -v kernel) ↩`

Freshen

- Install a new version of a package only if an older one is already installed, otherwise do nothing.

Erasing

- Remove a package:

```
$ sudo rpm -e ppracer ↩
```

Verfying

- Checking the files in a package all match the original data
- If it doesn't match, then flags show this:

flag	*what it means*
S	file *S*ize differs
M	*M*ode differs (includes permissions and file type)
5	MD*5* sum differs
D	*D*evice major/minor number mismatch
L	symbolic *L*ink points to wrong place
U	*U*ser ownership differs
G	*G*roup ownership differs
T	m*T*ime differs
C	selinux security *C*ontext differs

```
$ $ rpm -V setup ↩
S.5....T.  c /etc/aliases
S.5....TC  c /etc/bashrc
........C  c /etc/exports
S.5....T.  c /etc/printcap
S.5....TC  c /etc/profile
S.?....T.  c /etc/securetty
```

Checking Digital Signatures

- First, you need to *import* the public keys that are provided by Fedora:

  $ **sudo rpm --import /etc/pki/rpm-gpg/RPM-GPG-KEY*** ↩

- $ **rpm -K** ⟨*package-file.rpm*⟩ ↩

RPM Command Examples

command	effect
`rpm -qa | less`	list all installed software packages
`rpm -q httpd`	show the version of the httpd package, if it is installed
`rpm -qa | grep httpd`	show all installed packages that have *httpd* in their name
`rpm -ql httpd`	***l***ist all files in the httpd package
`rpm -qd httpd`	list all ***d***ocumentation files in the httpd package
`rpm -qc httpd`	list all ***c***onfiguration files in the httpd package
`rpm -qi httpd`	display ***i***nformation about the package
`rpm -V httpd`	***v***erify that the `httpd` package is correctly installed
`rpm -qf /etc/passwd`	determine which package the `/etc/passwd` ***f***ile belongs to

A quick comparison with `dpkg`

command	effect
`dpkg -list | less`	list all installed software packages
`dpkg -l httpd`	show the version of the `httpd` package, if it is installed
`dpkg -list | grep httpd`	show all installed packages that have *httpd* in their name
`dpkg -listfiles httpd`	list all files in the `httpd` package
`dpkg -print-avail httpd`	display information about the package
`dpkg -S /etc/passwd`	determine which package the `/etc/passwd` file belongs to

Yum

- Yum is a high level wrapper for RPM
- Uses the information in the RPM packages to figure out what things to install
- downloads the files you need, and installs them, lovingly taking care of which packages depend on what.

Why not stick with RPM?

- RPM is good, but it's too low level to easily manage all the packages on the system
- Yum uses the information in the RPM database, and the RPM files themselves to make sure that each software package has everything it needs.

What's all this primative text stuff?

- There are many GUI programs that use yum as their backend
- I have never used any of them, so your exercise is to Google for them!

Updates

- Software has bugs
- Fedora particularly has many new versions of software that are being made available, since it aims to provide the current version of most pieces of software.
- Fedora has more updates than any other stable distribution I know of.
- Only a few are for security reasons, most are for updates.

Why update? Why not update?

Why update?

- The easiest systems to crack have no updates.

Why not update?

- Updating can break a working system
- But that doesn't stop me from keeping my systems updated!
- . . . but not good if your business loses thousands of dollars per second of downtime.
  - In such cases, you have a test system where you test the updates first, check they don't stop your applications from working

How to update

Do this once only:

- Ensure your own account is in the `sudoers` file
  - see the `sudo` handout
- set your `http_proxy` environment variable
  - See the next section: 39 on the following page

Do this for each update:

- then do:
  `$ sudo yum -y update` ↩
- and watch it go.

Setting `http_proxy`

If access to web servers on the Internet is blocked by a firewall, and a proxy server is required, such as here in the TAFE:

- You need to set the `http_proxy` environment variable.
- The proxy server here has the DNS name "`gw`", and listens on port 5865 . . .
- . . . so set the `http_proxy` like this:
  `export http_proxy=http://gw:5865/`
- The best thing to do is to edit your login script `~/.bash_profile` and add it at the end:
  `emacs ~/.bash_profile &`
- Then log out and log in. . .
- You should be able to see the value now:
  `$ echo $http_proxy` ↩
  `http://gw:5865/`

The “repos”: repositories

- In the `/etc/yum.repos.d` directory are some files that end with “`.repo`”
- You can change these, add other repos (such as dag, freshrpms, atrpms, livna)
- You can also change the `baseurl` to a local Australian mirror (see `http://fedora.redhat.com/Download/mirrors.html` for a list of Australian mirrors)

Adding extras

- Well, the fedora extras repository is already enabled, so nothing to do here.

- If you add the `dag`, `freshrpms`, `atrpms` or `livna` repositories, you should import their public keys, and enable checking the GPG signatures of the packages.
- For example, here is my `/etc/yum.repos.d/dag.repos` file:

```
[dag]
name=Dag RPM Repository for Fedora Core
baseurl=http://apt.sw.be/fedora/$releasever/en/$basearc
enabled=1
gpgcheck=1
```

Resources

Dr. Peter Salus.
The Daemon, the GNU & the Penguin.
Grocklaw 2006.
http://www.groklaw.net/staticpages/index.php?page=20051013231901859.

Eric Foster-Johnson.
RPM Guide.
http://fedora.redhat.com/docs/drafts/rpm-guide-en/

The RPM Package Manager.
http://rpm.org/

Wikipedia entry.
Yellow dog Updater, Modified.
http://en.wikipedia.org/wiki/Yellow_dog_Updater,_Modified

Topics Covered

License Of This Document