第12章 安装RPM包或者安装源码包

在windows下安装一个软件很轻松，只要双击.exe的文件，安装提示连续“下一步”即可，然而linux系统下安装一个软件似乎并不那么轻松了，因为我们不是在图形界面下。所以你要学会如何在linux下安装一个软件。

在前面的内容中多次提到的yum，这个yum是Redhat所特有的安装RPM程序包的工具，使用起来相当方便。因为使用RPM安装某一个程序包有可能会因为该程序包依赖另一个程序包而无法安装。而使用yum工具就可以连同依赖的程序包一起安装。当然CentOS同样可以使用yum工具，而且在CentOS中你可以免费使用yum，但Redhat中只有当你付费后才能使用yum，默认是无法使用yum的。在介绍yum之前先说一说RPM相关的东西。

RPM工具

RPM是“Redhat Package Manager”的缩写，根据名字也能猜到这是Redhat公司开发出来的。RPM 是以一种数据库记录的方式来将你所需要的套件安装到你的Linux 主机的一套管理程序。也就是说，你的linux系统中存在着一个关于RPM的数据库，它记录了安装的包以及包与包之间依赖相关性。RPM包是预先在linux机器上编译好并打包好的文件，安装起来非常快捷。但是也有一些缺点，比如安装的环境必须与编译时的环境一致或者相当；包与包之间存在着相互依赖的情况；卸载包时需要先把依赖的包卸载掉，如果依赖的包是系统所必须的，那就不能卸载这个包，否则会造成系统崩溃。

如果你的光驱中还有系统安装盘的话，我们可以通过 `mount /dev/cdrom /mnt` 命令把光驱挂载到/mnt目录下，那么你会在/mnt/Packages目录下看到很多.rpm的文件，这就是RPM包了。

```
[root@localhost ~]# mount /dev/cdrom /mnt/
mount: block device /dev/sr0 is write-protected, mounting read-only
[root@localhost ~]# ls /mnt/
CentOS_BuildTag  Packages                    RPM-GPG-KEY-CentOS-Security-6
EULA             RELEASE-NOTES-en-US.html    RPM-GPG-KEY-CentOS-Testing-6
GPL              repodata                    TRANS.TBL
images           RPM-GPG-KEY-CentOS-6
isolinux         RPM-GPG-KEY-CentOS-Debug-6
[root@localhost ~]# ls /mnt/Packages/|head
389-ds-base-1.2.11.15-11.el6.i686.rpm
389-ds-base-libs-1.2.11.15-11.el6.i686.rpm
abrt-2.0.8-15.el6.centos.i686.rpm
abrt-addon-ccpp-2.0.8-15.el6.centos.i686.rpm
abrt-addon-kerneloops-2.0.8-15.el6.centos.i686.rpm
abrt-addon-python-2.0.8-15.el6.centos.i686.rpm
abrt-cli-2.0.8-15.el6.centos.i686.rpm
abrt-desktop-2.0.8-15.el6.centos.i686.rpm
abrt-gui-2.0.8-15.el6.centos.i686.rpm
abrt-libs-2.0.8-15.el6.centos.i686.rpm
```

每一个rpm包的名称都由 `-` 和 `.` 分成了若干部分。就拿“abrt-cli-2.0.8-15.el6.centos.i686.rpm”这个包来解释一下，“abrt-cli”为包名，“2.0.8”则为版本信息，“15.el6.centos”为发布版本号，“i686”为运行平台。其中运行平台常见的有i386，i586，i686，x86_64 ，需要你注意的是cpu目前是分32位和64位的，i386,i586和i686都为32位平台，x86_64则代表为64位的平台。另外有些rpm包并没有写具体的平台而是noarch，这代表这个rpm包没有硬件平台限制。例如 “alacarte-0.10.0-1.fc6.noarch.rpm”. 下面介绍一下rpm常用的命令。

1. 安装一个rpm包

```
[root@localhost ~]# rpm -ivh /mnt/Packages/libjpeg-turbo-devel-1.2.1-1.el6.i686.rpm
Preparing...                ########################################### [100%]
   1:libjpeg-turbo-devel    ########################################### [100%]
```

“-i”：安装的意思

“-v”：可视化

“-h”：显示安装进度

另外在安装一个rpm包时常用的附带参数有：

`--force`：强制安装，即使覆盖属于其他包的文件也要安装

`--nodeps`：当要安装的rpm包依赖其他包时，即使其他包没有安装，也要安装这个包

2. 升级一个rpm包

命令 `rpm -Uvh filename`

“-U”：即升级的意思

3. 卸载一个rpm包

命令 `rpm -e filename`

这里的filename是通过rpm的查询功能所查询到的，稍后会作介绍。

```
[root@localhost ~]# rpm -qa |grep libjpeg-turbo-devel
libjpeg-turbo-devel-1.2.1-1.el6.i686
[root@localhost ~]# rpm -e libjpeg-turbo-devel
```

卸载时后边跟的filename和安装时的是有区别的，安装时是把一个存在的文件作为参数，而卸载时只需要包名即可。

4. 查询一个包是否安装

目录列表

SEARCH

Go

Enter search terms or a module, class or function name.

命令 `rpm -q rpm包名` (这里的包名，是不带有平台信息以及后缀名的)

```
[root@localhost ~]# rpm -q libjpeg-turbo-devel
package libjpeg-turbo-devel is not installed
[root@localhost ~]# rpm -ivh /mnt/Packages/libjpeg-turbo-devel-1.2.1-1.el6.i686.rpm
Preparing...                ########################################### [100%]
   1:libjpeg-turbo-devel    ########################################### [100%]
[root@localhost ~]# rpm -q libjpeg-turbo-devel
libjpeg-turbo-devel-1.2.1-1.el6.i686
```

我们可以使用 `rpm -qa` 查询当前系统所有安装过的rpm包，限于篇幅，阿铭只列出前十个。

```
[root@localhost ~]# rpm -qa |head
plymouth-core-libs-0.8.3-27.el6.centos.i686
xml-common-0.6.3-32.el6.noarch
sgpio-1.2.0.10-5.el6.i686
iso-codes-3.16-2.el6.noarch
gnome-vfs2.2.27.2-6.el6.i686
libX11-common-1.5.0-4.el6.noarch
curl-7.19.7-35.el6.i686
ca-certificates-2010.63-3.el6_1.5.noarch
cups-libs-1.4.2-48.el6_3.3.i686
kbd-misc-1.15-11.el6.noarch
```

5．得到一个已安装rpm包的相关信息

命令 `rpm -qi 包名` （同样不需要加平台信息与后缀名）

```
[root@localhost ~]# rpm -qi libjpeg-turbo-devel
Name        : libjpeg-turbo-devel          Relocations: (not relocatable)
Version     : 1.2.1                             Vendor: CentOS
Release     : 1.el6                         Build Date: 2013年02月22日 星期五 06时49分08秒
Install Date: 2013年05月13日 星期一 01时37分48秒      Build Host: c6b9.bsys.dev.centos.org
Group       : Development/Libraries         Source RPM: libjpeg-turbo-1.2.1-1.el6.src.rpm
Size        : 321085                           License: wxWidgets
Signature   : RSA/SHA1, 2013年02月24日 星期日 01时53分55秒, Key ID 0946fca2c105b9de
Packager    : CentOS BuildSystem <http://bugs.centos.org>
URL         : http://sourceforge.net/projects/libjpeg-turbo
Summary     : Headers for the libjpeg-turbo library
Description :
This package contains header files necessary for developing programs which
will manipulate JPEG files using the libjpeg-turbo library.
```

6．列出一个rpm包安装的文件

命令 `rpm -ql 包名`

```
[root@localhost ~]# rpm -ql libjpeg-turbo-devel
/usr/include/jconfig.h
/usr/include/jerror.h
/usr/include/jmorecfg.h
/usr/include/jpeglib.h
/usr/lib/libjpeg.so
/usr/share/doc/libjpeg-turbo-devel-1.2.1
/usr/share/doc/libjpeg-turbo-devel-1.2.1/coderules.txt
/usr/share/doc/libjpeg-turbo-devel-1.2.1/example.c
/usr/share/doc/libjpeg-turbo-devel-1.2.1/jconfig.txt
/usr/share/doc/libjpeg-turbo-devel-1.2.1/libjpeg.txt
/usr/share/doc/libjpeg-turbo-devel-1.2.1/structure.txt
```

通过上面的命令可以看出文件“/usr/lib/libjpeg.so”是通过安装“libjpeg-turbo-devel”这个rpm包得来的。那么反过来如何通过一个文件去查找是由安装哪个rpm包得来的？

7．列出某一个文件属于哪个rpm包

命令 `rpm -qf 文件的绝对路径`

```
[root@localhost ~]# rpm -qf /usr/lib/libjpeg.so
libjpeg-turbo-devel-1.2.1-1.el6.i686
```

yum工具

在前面的章节中，阿铭多次提到yum工具，今天终于该讲它了。这个工具比rpm工具好用多了，当然前提是你使用的linux系统是支持yum的。yum最大的优势在于可以联网去下载所需要的rpm包，然后自动安装，在这个工程中如果要安装的rpm包有依赖关系，yum会帮你解决掉这些依赖关系依次安装所有rpm包。下面阿铭介绍常用的yum 命令。

1．列出所有可用的rpm包 “yum list”

```
[root@localhost ~]# yum list |head -n 20
Loaded plugins: fastestmirror, security
Loading mirror speeds from cached hostfile
 * base: mirrors.btte.net
 * extras: mirrors.btte.net
 * updates: mirrors.btte.net
Installed Packages
ConsoleKit.i686                         0.4.1-3.el6               @anaconda-CentOS-201303020136.i386/6.4
ConsoleKit-libs.i686                    0.4.1-3.el6               @anaconda-CentOS-201303020136.i386/6.4
ConsoleKit-x11.i686                     0.4.1-3.el6               @anaconda-CentOS-201303020136.i386/6.4
GConf2.i686                             2.28.0-6.el6              @anaconda-CentOS-201303020136.i386/6.4
MAKEDEV.i686                            3.24-6.el6                @anaconda-CentOS-201303020136.i386/6.4
ORBit2.i686                             2.14.17-3.2.el6_3         @anaconda-CentOS-201303020136.i386/6.4
abrt.i686                               2.0.8-15.el6.cento        @anaconda-CentOS-201303020136.i386/6.4
abrt-addon-ccpp.i686                    2.0.8-15.el6.cento        @anaconda-CentOS-201303020136.i386/6.4
```

```
abrt-addon-kerneloops.i686              2.0.8-15.el6.cento         @anaconda-CentOS-201303020136.i386/6.4
abrt-addon-python.i686                  2.0.8-15.el6.cento         @anaconda-CentOS-201303020136.i386/6.4
abrt-cli.i686                           2.0.8-15.el6.centos        @anaconda-CentOS-201303020136.i386/6.4
abrt-libs.i686                          2.0.8-15.el6.centos        @anaconda-CentOS-201303020136.i386/6.4
abrt-tui.i686                           2.0.8-15.el6.centos        @anaconda-CentOS-201303020136.i386/6.4
acl.i686                                2.2.49-6.el6               @anaconda-CentOS-201303020136.i386/6.4
```

限于篇幅，阿铭只列举出来前14个包信息。从上例中可以看到有“mirrors.btte.net”信息出现，这是在告诉用户，它是从mirrors.btte.net这里下载到的rpm包资源。从上面的例子中你还可以看到最左侧是rpm包名字，中间是版本信息，最右侧是安装信息，如果安装了就显示类似“@anaconda-CentOS”，“@base”或者“@extras”，他们前面都会有一个“@”符号，这很好区分。未安装则显示base或者extras，如果是该rpm包已安装但需要升级则显示updates. 如果你看的仔细会发现，“yum list”会先列出已经安装的包(Installed Packages)然后再列出可以安装的包(Available Packages)

2. 搜索一个rpm包

命令 `yum search [相关关键词]`

```
[root@localhost ~]# yum search vim
Loaded plugins: fastestmirror, security
Loading mirror speeds from cached hostfile
 * base: mirrors.btte.net
 * extras: mirrors.btte.net
 * updates: mirrors.btte.net
============================================ N/S Matched: vim ============================================
vim-X11.i686 : The VIM version of the vi editor for the X Window System
vim-common.i686 : The common files needed by any version of the VIM editor
vim-enhanced.i686 : A version of the VIM editor which includes recent enhancements
vim-minimal.i686 : A minimal version of the VIM editor

  Name and summary matches only, use "search all" for everything.
```

除了这样搜索外，阿铭常用的是利用grep来过滤

```
[root@localhost ~]# yum list |grep 'vim'
vim-common.i686                     2:7.2.411-1.8.el6          @anaconda-CentOS-201303020136.i386/6.4
vim-enhanced.i686                   2:7.2.411-1.8.el6          @anaconda-CentOS-201303020136.i386/6.4
vim-minimal.i686                    2:7.2.411-1.8.el6          @anaconda-CentOS-201303020136.i386/6.4
vim-X11.i686                        2:7.2.411-1.8.el6          base
```

我们同样可以找到相应的rpm包。

3. 安装一个rpm包

命令 `yum install [-y] [rpm包名]`

如果不加“-y”选项，则会以与用户交互的方式安装，首先是列出需要安装的rpm包信息，然后会问用户是否需要安装，输入y则安装，输入n则不安装。而阿铭嫌这样太麻烦，所以直接加上“-y”选项，这样就省略掉了问用户是否安装的那一步。

```
[root@localhost ~]# yum install vim-X11
Loaded plugins: fastestmirror, security
Loading mirror speeds from cached hostfile
 * base: mirrors.btte.net
 * extras: mirrors.btte.net
 * updates: mirrors.btte.net
Setting up Install Process
Resolving Dependencies
--> Running transaction check
---> Package vim-X11.i686 2:7.2.411-1.8.el6 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package          Arch          Version                    Repository     Size
================================================================================
Installing:
 vim-X11          i686          2:7.2.411-1.8.el6          base          975 k

Transaction Summary
================================================================================
Install       1 Package(s)

Total download size: 975 k
Installed size: 2.1 M
Is this ok [y/N]: y
Downloading Packages:
vim-X11-7.2.411-1.8.el6.i686.rpm                          | 975 kB     00:03
Running rpm_check_debug
Running Transaction Test
Transaction Test Succeeded
Running Transaction
  Installing : 2:vim-X11-7.2.411-1.8.el6.i686                              1/1
  Verifying  : 2:vim-X11-7.2.411-1.8.el6.i686                              1/1

Installed:
  vim-X11.i686 2:7.2.411-1.8.el6

Complete!
```

4. 卸载一个rpm包

命令 `yum remove [-y] [rpm包名]`

```
[root@localhost ~]# yum remove vim-X11
```

```
Loaded plugins: fastestmirror, security
Setting up Remove Process
Resolving Dependencies
--> Running transaction check
---> Package vim-X11.i686 2:7.2.411-1.8.el6 will be erased
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package          Arch          Version                    Repository      Size
================================================================================
Removing:
 vim-X11          i686          2:7.2.411-1.8.el6          @base          2.1 M

Transaction Summary
================================================================================
Remove        1 Package(s)

Installed size: 2.1 M
Is this ok [y/N]: y
Downloading Packages:
Running rpm_check_debug
Running Transaction Test
Transaction Test Succeeded
Running Transaction
  Erasing    : 2:vim-X11-7.2.411-1.8.el6.i686                               1/1
  Verifying  : 2:vim-X11-7.2.411-1.8.el6.i686                               1/1

Removed:
  vim-X11.i686 2:7.2.411-1.8.el6

Complete!
```

卸载和安装一样，也可以直接加上“-y”选项来省略掉和用户交互的步骤。在这里阿铭要提醒一下，卸载某个rpm包一定要看清楚了，不要连其他重要的rpm包一起卸载了，以免影响正常的业务。

4. 升级一个rpm包

命令 `yum update [-y] [rpm包]`

```
[root@localhost ~]# yum update libselinux
Loaded plugins: fastestmirror, security
Loading mirror speeds from cached hostfile
 * base: mirrors.btte.net
 * extras: mirrors.btte.net
 * updates: mirrors.btte.net
Setting up Update Process
Resolving Dependencies
--> Running transaction check
---> Package libselinux.i686 0:2.0.94-5.3.el6 will be updated
--> Processing Dependency: libselinux = 2.0.94-5.3.el6 for package: libselinux-utils-2.0.94-5.3.el6.i686
---> Package libselinux.i686 0:2.0.94-5.3.el6_4.1 will be an update
--> Running transaction check
---> Package libselinux-utils.i686 0:2.0.94-5.3.el6 will be updated
---> Package libselinux-utils.i686 0:2.0.94-5.3.el6_4.1 will be an update
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package                Arch       Version                  Repository     Size
================================================================================
Updating:
 libselinux             i686       2.0.94-5.3.el6_4.1       updates       108 k
Updating for dependencies:
 libselinux-utils       i686       2.0.94-5.3.el6_4.1       updates        81 k

Transaction Summary
================================================================================
Upgrade       2 Package(s)

Total download size: 189 k
Is this ok [y/N]: y
Downloading Packages:
(1/2): libselinux-2.0.94-5.3.el6_4.1.i686.rpm            | 108 kB     00:00
(2/2): libselinux-utils-2.0.94-5.3.el6_4.1.i686.rpm      |  81 kB     00:00
--------------------------------------------------------------------------------
Total                                        1.0 MB/s | 189 kB     00:00
Running rpm_check_debug
Running Transaction Test
Transaction Test Succeeded
Running Transaction
  Updating   : libselinux-2.0.94-5.3.el6_4.1.i686                           1/4
  Updating   : libselinux-utils-2.0.94-5.3.el6_4.1.i686                     2/4
  Cleanup    : libselinux-utils-2.0.94-5.3.el6.i686                         3/4
  Cleanup    : libselinux-2.0.94-5.3.el6.i686                               4/4
  Verifying  : libselinux-utils-2.0.94-5.3.el6_4.1.i686                     1/4
  Verifying  : libselinux-2.0.94-5.3.el6_4.1.i686                           2/4
  Verifying  : libselinux-2.0.94-5.3.el6.i686                               3/4
  Verifying  : libselinux-utils-2.0.94-5.3.el6.i686                         4/4

Updated:
  libselinux.i686 0:2.0.94-5.3.el6_4.1
```

```
Dependency Updated:
  libselinux-utils.i686 0:2.0.94-5.3.el6_4.1

Complete!
```

以上介绍了如何使用yum搜索、安装、卸载以及升级一个rpm包，如果你掌握了这些那么你就已经可以解决日常工作中遇到的与rpm包相关问题了。当然yum工具还有好多其他好用的命令，阿铭不再列举出来，如果你感兴趣就去man 一下吧。除此之外，阿铭还会教你一些关于yum的小应用。

使用本地的光盘来制作一个yum源

有时候你的linux系统不能联网，当然就不能很便捷的使用联网的yum源了，这时候就需要你自己会利用linux系统光盘制作一个yum源。具体步骤如下：

a）挂载光盘

```
[root@localhost ~]# mount /dev/cdrom /mnt
```

b）删除/etc/yum.repos.d目录所有的repo文件

```
[root@localhost ~]# rm -rf /etc/yum.repos.d/*
```

c） 创建新文件dvd.repo

```
[root@localhost ~]# vim /etc/yum.repos.d/dvd.repo
```

加入以下内容：

```
[dvd]
name=install dvd
baseurl=file:///mnt
enabled=1
gpgcheck=0
```

d） 刷新 repos 生成缓存

```
[root@localhost ~]# yum makecache
```

然后就可以使用yum命令安装你所需要的软件包了

利用yum工具下载一个rpm包

有时，我们需要下载一个rpm包，只是下载下来，拷贝给其他机器使用，前面也介绍过yum安装rpm包的时候，首先得下载这个rpm包然后再去安装，所以使用yum完全可以做到只下载而不安装。

a）首先要安装 yum-downloadonly

```
[root@localhost ~]# yum install -y yum-plugin-downloadonly.noarch
```

如果你的CentOS是5.x版本，则需要安装yum-downloadonly.noarch这个包。

b）下载一个rpm包而不安装

```
[root@localhost ~]# yum install 包名 -y --downloadonly
```

这样虽然下载了，但是并没有保存到我们想要的目录下，那么如何指定目录呢？

c）下载到指定目录

```
[root@localhost ~]# yum install 包名 -y --downloadonly --downloaddir=/usr/local/src
```

下面阿铭下载一个rpm包：

```
[root@localhost ~]# yum install -y yum-presto.noarch  --downloadonly --downloaddir=/usr/local/src/
Loaded plugins: downloadonly, fastestmirror, security
Loading mirror speeds from cached hostfile
 * base: mirrors.btte.net
 * extras: mirrors.btte.net
 * updates: mirrors.btte.net
Setting up Install Process
Resolving Dependencies
--> Running transaction check
---> Package yum-presto.noarch 0:0.6.2-1.el6 will be installed
--> Processing Dependency: deltarpm >= 3.4-2 for package: yum-presto-0.6.2-1.el6.noarch
--> Running transaction check
---> Package deltarpm.i686 0:3.5-0.5.20090913git.el6 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package          Arch         Version                       Repository  Size
================================================================================
Installing:
 yum-presto       noarch       0.6.2-1.el6                   base        32 k
Installing for dependencies:
 deltarpm         i686         3.5-0.5.20090913git.el6       base        73 k

Transaction Summary
================================================================================
Install       2 Package(s)

Total download size: 105 k
Installed size: 257 k
Downloading Packages:
--------------------------------------------------------------------------------
Total                                           43 MB/s | 105 kB     00:00


exiting because --downloadonly specified
```

```
[root@localhost ~]# ls /usr/local/src/
deltarpm-3.5-0.5.20090913git.el6.i686.rpm  yum-presto-0.6.2-1.el6.noarch.rpm
```

安装源码包

其实，在linux下面安装一个源码包是最常用的，阿铭在日常的管理工作中，大部分软件都是通过源码安装的。安装一个源码包，是需要我们自己把源代码编译成二进制的可执行文件。如果你读得懂这些源代码，那么你就可以去修改这些源代码自定义功能，然后再去编译成你想要的。使用源码包的好处除了可以自定义修改源代码外还可以定制相关的功能，因为源码包在编译的时候是可以附加额外的选项的。

源码包的编译用到了linux系统里的编译器，常见的源码包一般都是用C语言开发的，这也是因为C语言为linux上最标准的程序语言。Linux上的C语言编译器叫做gcc，利用它就可以把C语言变成可执行的二进制文件。所以如果你的机器上没有安装gcc就没有办法去编译源码。你可以使用 `yum install -y gcc` 来完成安装。

安装一个源码包，通常需要三个步骤：

1）./configure

在这一步可以定制功能，加上相应的选项即可，具有有什么选项可以通过 `./configure --help` 命令来查看。在这一步会自动检测你的linux系统与相关的套件是否有编译该源码包时需要的库，因为一旦缺少某个库就不能完成编译。只有检测通过后才会生成一个Makefile文件。

2） **make**

使用这个命令会根据Makefile文件中预设的参数进行编译，这一步其实就是gcc在工作了。

3） **make install**

安装步骤，生成相关的软件存放目录和配置文件的过程。

上面介绍的3步并不是所有的源码包软件都一样的，阿铭以前也曾经遇到过，安装步骤并不是这样，也就是说源码包的安装并非具有一定的标准安装步骤。这就需要你拿到源码包解压后，然后进入到目录找相关的帮助文档，通常会以INSTALL或者README为文件名。所以，你一定要去看一下。下面阿铭会编译安装一个源码包来帮你更深刻的去理解如何安装源码包。

1. 下载一个源码包

下载源码包一定要去官方站点去下载，不要在网上随便下载，那样很不安全。因为你下载到的源码包很有可能是被人修改过的。

```
[root@localhost src]# cd /usr/local/src/
[root@localhost src]# wget http://mirrors.hust.edu.cn/apache/httpd/httpd-2.2.27.tar.bz2
```

阿铭提供的下载地址为apache官方网站上提供的一个镜像，下载速度还可以。在下载之前，阿铭进入到了“/usr/local/src”目录，这是因为阿铭习惯把源码包都放到这个目录下，这样做的好处是，方便自己和其他管理员维护，所以阿铭给你一个建议，以后下载的源码包都统一放到这个目录下吧。

2. 解压源码包

```
[root@localhost src]# tar jxvf httpd-2.2.27.tar.bz2
```

3. 配置相关的选项，并生成Makefile

```
[root@localhost src]# cd httpd-2.2.27
[root@localhost httpd-2.2.27]# ./configure --help |less
`configure' configures this package to adapt to many kinds of systems.

Usage: ./configure [OPTION]... [VAR=VALUE]...

To assign environment variables (e.g., CC, CFLAGS...), specify them as
VAR=VALUE.  See below for descriptions of some of the useful variables.

Defaults for the options are specified in brackets.

Configuration:
  -h, --help              display this help and exit
      --help=short        display options specific to this package
      --help=recursive    display the short help of all the included packages
  -V, --version           display version information and exit
  -q, --quiet, --silent   do not print `checking ...' messages
      --cache-file=FILE   cache test results in FILE [disabled]
  -C, --config-cache      alias for `--cache-file=config.cache'
  -n, --no-create         do not create output files
      --srcdir=DIR        find the sources in DIR [configure dir or `..']
```

后面的内容阿铭省略掉了，阿铭使用 `./configure --help` 命令查看可以使用的选项。一般常用的有 `--prefix=PREFIX` 这个选项的意思是定义软件包安装到哪里。到这里，阿铭再提一个小小的建议，通常源码包都是安装在/usr/local/目录下的。比如，我们把apache安装在/usr/local/apache2下，那么这里就应该这样写 `--prefix=/usr/local/apache2` 其他还有好多选项，如果你有耐心可以挨个去看一看都有什么作用。

```
[root@localhost httpd-2.2.27]# ./configure --prefix=/usr/local/apache2
checking for chosen layout... Apache
checking for working mkdir -p... yes
checking build system type... i686-pc-linux-gnu
checking host system type... i686-pc-linux-gnu
checking target system type... i686-pc-linux-gnu

Configuring Apache Portable Runtime library ...

checking for APR... reconfig
configuring package in srclib/apr now
checking build system type... i686-pc-linux-gnu
checking host system type... i686-pc-linux-gnu
checking target system type... i686-pc-linux-gnu
Configuring APR library
Platform: i686-pc-linux-gnu
checking for working mkdir -p... yes
APR Version: 1.4.6
```

```
checking for chosen layout... apr
checking for gcc... no
checking for cc... no
checking for cl.exe... no
configure: error: in `/usr/local/src/httpd-2.2.27/srclib/apr':
configure: error: no acceptable C compiler found in $PATH
See `config.log' for more details
configure failed for srclib/apr
```

不幸的是，阿铭一开始就报错了，因为没有gcc编译器，需要先安装一下。

```
[root@localhost httpd-2.2.27]# yum install -y gcc
```

由于gcc依赖的包很多，所以安装时间会长一些。安装完后，再继续上面的步骤。

tcode：

```
[root@localhost httpd-2.2.27]# ./configure --prefix=/usr/local/apache2
```

验证这一步是否成功的命令是：

```
[root@localhost httpd-2.2.27]# echo $?
0
```

返回值如果是“0”则执行成功，否则就是没有成功。此时就成功生成 Makefile 了。

```
[root@localhost httpd-2.2.27]# ls -l Makefile
-rw-r--r-- 1 root root 8954 5月  13 12:02 Makefile
```

4. 进行编译

```
[root@localhost httpd-2.2.27]# make
-bash: make: command not found
```

又发生错误了，提示“make”命令没有发现，解决办法是安装make工具。

```
[root@localhost httpd-2.2.27]# yum install -y make
```

继续make

```
[root@localhost httpd-2.2.27]# make
Making all in srclib
make[1]: Entering directory `/usr/local/src/httpd-2.2.27/srclib'
Making all in apr
make[2]: Entering directory `/usr/local/src/httpd-2.2.27/srclib/apr'
make[3]: Entering directory `/usr/local/src/httpd-2.2.27/srclib/apr'
/bin/sh /usr/local/src/httpd-2.2.27/srclib/apr/libtool --silent --mode=compile gcc -g -O2 -pthread   -DHAVE_CONFIG_H -DLINUX=2 -D_REENTRANT -D_GNU_SOURCE -D_LA
```

编译的时候，就会出现类似这么多乱七八糟的信息，编译的时间比较长，CPU使用率会很高，这是因为CPU高速计算，编译完后，再使用 `echo $?` 验证一下是否正常成功。

```
[root@localhost httpd-2.2.27]# echo $?
0
```

如果是0的话，就可以执行最后一步了。

5. 安装

```
[root@localhost httpd-2.2.27]# make install
Making install in srclib
make[1]: Entering directory `/usr/local/src/httpd-2.2.27/srclib'
Making install in apr
make[2]: Entering directory `/usr/local/src/httpd-2.2.27/srclib/apr'
make[3]: Entering directory `/usr/local/src/httpd-2.2.27/srclib/apr'
make[3]: Nothing to be done for `local-all'.
make[3]: Leaving directory `/usr/local/src/httpd-2.2.27/srclib/apr'
```

当然你也可以使用 `echo $?` 看看有没有正确安装，执行完这一步，则会在“/usr/local/apache2”目录下增加了很多目录。

```
[root@localhost httpd-2.2.27]# ls /usr/local/apache2/
bin    cgi-bin  error   icons    lib   man     modules
build  conf     htdocs  include  logs  manual
```

到此，apache源码的安装就完成了，其实在日常的源码安装工作中，并不是谁都像阿铭这样顺利完成安装的，遇到错误不能完成安装的情况是很多的。通常都是因为缺少某一个库文件导致的。这就需要你仔细琢磨报错信息或者查看当前目录下的 “config.log” 去得到相关的信息。另外，如果自己不能解决那就去网上google一下吧，通常你会得到想要的答案。

阿铭建议你最好再扩展学习一下： http://www.aminglinux.com/bbs/thread-5436-1-1.html

教程答疑：请移步这里.

欢迎你加入 阿铭学院 和阿铭一起学习Linux，让阿铭成为你Linux生涯中永远的朋友吧！