第4章 安装软件

因为周围有太多的系统管理中心需要安装提供各种服务所必须的软件，因此学习新软件包的安装机制和学习如何编译以源代码形式传播的软件包就是十分重要的了。

一般的情况下，大多数应用程序都有非常相似的安装操作模式。密切注意信息的主要来源，再加上一些常识，就可以使大多数的安装过程顺利进行。因此，本章的主题不仅要介绍几类软件包的安装方案，还将介绍一些故障处理策略。

注意 本书中讨论的全部软件包都已经通过了测试，证明它们都能够工作在 Linux操作系统中，都能够顺利地通过编译和安装。安装过程应该不会出现问题，但是版本变化和不同的硬件配置可能会引起软件的崩溃。学习和掌握总是要比坐等好。

本章将讨论两种最常用的软件安装方法：使用 Red Hat Package Manager软件包管理工具软件（RPM）和自行编译源代码。本章中所有需要输入的命令都是以根用户的身份输入的。因此最好是直接以根用户身份登录进入系统（第 6章将告诉读者如何使用 su命令把自己的用户ID修改为其他人的ID）。一般情况下，如果读者不是以根用户身份登录进入系统的话，则需要先打开一个终端窗口，然后在提示符处输入 su -root命令。

4.1 Red Hat Package Manager软件包管理工具软件

Red Hat Package Manager（RPM）软件包管理工具软件的基本功能是安装和清除文件（通常是对已经预编译好的软件进行操作）。它使用起来很简便，而一些以它为核心建立的图形化操作接口使用起来就更简便了。 Red Hat、Caldera和其他发行版本都已经开始使用这个工具软件发行他们的软件。事实上，本书涉及的软件几乎都有 RPM格式的版本。我们在本章中学习自行编译软件步骤的目的是为了给软件加上时间选项，而这是 RPM软件包所不具备的。

从根本上讲，一个 RPM文件是能够让某个特定程序运行的全部文件的一个集合。它还包括对程序的说明、版本信息以及实现安装过程本身必须的脚本程序等。

RPM工具对安装在某个指定主机上的全部 RPM软件包进行全面的管理。包括已经安装了哪些软件包、它们的版本号码以及文件的存放位置等方面的记录。这些资料全部保存在主机上的一个简单的数据库文件中。

一般来说，RPM格式的软件要比需要编译的软件更容易安装和维护。因为使用 RPM的时候用户也就接受了这个 RPM中提供的缺省参数。大多数情况下，这些缺省参数使用起来都不会有什么问题。但是如果用户需要更具体地关心某项服务的实现过程，那么自行编译源代码将会使用户对软件包中都有哪些组件以及它们是如何协调工作的有更进一步的了解。

如果读者只想安装一个简单的软件包的话， RPM就足够了。有几个地方是很不错的 RPM软件包来源，比如：

- http: //www.rpmfind.net
- ftp: //ftp.redhat.com/pub/contrib

• http: //www.linuxapps.com

当然，如果读者对 RPM本身的其他细节感兴趣，可以访问 RPM的互连网站点： http: //www.rpm.org。RPM是随Red Hat Linux（及其变体）和Caldera Linux两种发行版本一起提供的。如果读者不知道自己的发行版本中是否有 RPM，可以向供货商查询。

注意 虽然软件包的名称中有“ Red Hat ”，但是这个软件同样可以用在其他的发行版本中。事实上， RPM甚至已经被引入到其他的操作系统中去了，比如 Solaris和IRIX！RPM软件的源代码是开放的，这样任何人都可以编译系统使它们更好地工作。

4.1.1 安装新的软件包

安装一个新软件包最简便的方法就是使用 RPM的-i参数。举例来说，如果我们下载了一个名为bc-1.05a-4.i386.rpm的软件包并打算安装它，可以输入下面的命令：

```
[root@ford /root] # rpm -i bc-1.05a-4.i386.rpm
```

如果安装过程很顺利，我们根本看不到任何错误信息。这是安装 RPM软件包最常用的方法。如果这个软件已经安装过了，我们会看到如下所示的信息：

```
error : package bc-1.05a-4  is already installed .
```

有些软件包需要依赖于其他软件包。比如说一个游戏，可能就需要依赖于已经安装好的SVGA函数库。在这些情况下，读者会看到一些信息，告诉读者需要提前安装哪些软件包。只要安装好那些软件包再重新安装最初的软件包就可以了。

如果需要升级一个已经存在的软件包，需要使用 -U参数，如下所示：

```
[root@ford /root] # rpm -U bc-1.05a-4.i386.rpm
```

RPM的一些附加命令行参数列在表 4-1中。

举例来说，如果打算不理会依赖关系或者其他错误而强行安装某个软件包，则需要输入如下所示的命令：

```
[root@ford /root] # rpm -i --force -nodeps packagename.rpm
```

其中的packagename.rpm是需要安装的软件包的名称。

表4-1 RPM命令行参数

命令行参数	说 明
---force	强行安装。典型的使用情况是：准备安装某个奇怪或者不寻常的配置，但是RPM软件的安全措施禁止继续操作。 --force参数告诉RPM不做任何安全检查，只管安装就行。使用这个参数时一定要谨慎
-h	使用符号“ # ”指示安装进度，与-v参数一起使用时显示效果更好
--percent	显示已完成的百分比指示安装进度。如果从另外一个程序（比如某个 Perl脚本程序）中来运行RPM，并且想了解安装的进度时，这个参数就很方便
-nodeps	如果RPM提示丢失了依赖文件，但是读者打算无论如何也要完成安装，那么在命令行上使用这个参数将使RPM不进行依赖关系检查
-test	这个参数并不进行真正的安装；它只是用来检查安装能否成功地完成。如果发现了问题，会把问题打印到显示器屏幕上
-v	告诉RPM报告每一步操作的情况

4.1.2 查询软件包

有的时候，了解系统中都已经安装了哪些软件包以及它们的用途是很有用的， RPM的查询参数就可以做到这一点。

要想列出已经安装的全部软件包，请输入以下命令：

```
[root@ford /root] # rpm -qa
```

软件包清单太长了！如果读者正在查找一个特别的软件包名称，可以使用 grep命令指定软件包的名称（或者部分名称），如下所示：

```
[root@ford /root] # rpm -qa | grep -i 'name'
```

注意 grep中的-i参数告诉该命令区分大小写。

如果读者只是想每次一屏地查看全部的软件包，可以使用 more命令，如下所示：

```
[root@ford /root] # rpm -qa | more
```

要想找出某个特定的文件到底是属于哪个软件包的，可以输入：

```
[root@ford /root] # rpm -qf filename
```

其中的filename是准备要查找其归属的文件名称。

要想查出某个已经安装的软件包的功能，首先必须知道这个软件包的名称（从 rpm -qa的输出清单中找到），然后用如下所示的命令：

```
[root@ford /root] # rpm -qi packagename
```

其中的packagename是要查找其用途的软件包名称。

要想了解某个软件包中都有哪些文件，可以输入：

```
[root@ford /root] # rpm -qlp packagename
```

其中的packagename是准备要了解其组成文件的软件包名称。

4.1.3 反安装（清除）软件包

使用RPM反安装（清除）软件包就像安装它们一样简单。大多数情况下，用户只需要输入命令：

```
[root@ford /root] # rpm -e packagename
```

其中的packagename是rpm -qa命令清单中列出的软件包名称。

4.1.4 gnorpm工具

那些喜欢使用 GUI工具简化操作的人们可以使用 gnorpm软件。虽然它是被设计运行于GNOME环境的，但在KDE环境中也运行得相当好。它可以完成通过命令行完成的全部操作，但是用户就不必费力记住那些命令行参数了。当然，这样做是要增加一些系统开支的，这也是命令行版本依然存在的原因。

4.2 自行编译软件

源代码开放软件的重要优点之一是用户手里已经有了源代码。如果程序员不再开发了，用户还可以继续弄到。如果发现问题了，用户就可以修补它。换句话说，是用户控制着形势，

用不着再依靠某个用户无法控制的职业程序员了。但是有了源代码就意味着用户还必须能够对它进行编译，否则用户只不过是攥着一大把没什么用处的文本文件而已。

虽然本书涉及的每一个软件几乎都有对应的 RPM文件，我们仍准备对它进行一次编译，这样就可以选择并使用编译时间参数，而使用 RPM文件是做不到这一点的。所以没有必要害怕自行编译软件包。

在本小节中，我们将一个步骤一个步骤地对 SSH软件包进行编译，这个软件包是一个允许对远程主机进行安全连接的工具软件（ SSH在第17章中介绍）。SSH是一个典型的软件包；读者将会发现其他大多数需要进行编译的软件包差不多都遵守同样的通用格式。

4.2.1 获得并解压缩新的软件包

以源代码形式出现的软件通常都是一个“ tarball ”文件 —— 也就是说先是被归档为单独的一个大文件，再进行压缩。用来完成这个任务的是 tar命令和gzip命令：tar命令负责把许多文件归档为单独的一个大文件，而 gzip命令负责进行压缩。

注意 不要把gzip命令和WinZip软件弄混了。它们是两个不同的程序，使用两种不同的（但是差不多）方法进行压缩。需要注意的是 WinZip不知道怎样处理tarball文件。

典型情况下，人们通常都会选用一个独立完整的子目录对 tarball文件进行制作和保存工作。这样在需要从其中取出什么东西的时候，系统管理员就可以把各个软件包的 tarball文件保存到一个安全的位置。这还可以让系统管理员掌握除了基本的系统以外，机器里还安装有哪些软件包。/usr/local/src就是一个不错的子目录位置，因为站点上属于本机的软件一般都安装在/usr/local子目录中。

要想下载SSH的tarball文件，读者可以通过访问官方站点 http: //www.ssh.fi而得到它。

要想打开SSH的tarball文件，先把这个文件转移到/usr/local/src子目录中，如下所示：

```
[root@ford /root] # mv ssh-2.0.13.tar.gz /usr/local/src
```

注意 这里假设SSH的tarball文件的名字是ssh-2.0.13.tar.gz，并且已经被下载到/root子目录里。

接下来，使用cd命令把路径切换到/usr/local/src子目录去，如下所示：

```
[root@ford /root] # cd /usr/local/src
```

然后使用下面的命令解压缩tarball文件：

```
[root@ford src] # tar -xvzf ssh-2.0.13.tar.gz
```

上面tar命令中的z参数调用gzip命令在解归档操作之前先进行解压缩操作， v参数告诉tar命令在执行解归档操作的同时显示被处理文件的文件名。这样读者就可以知道源代码全部被解压缩之后存放在其中的子目录的名字了。我们现在应该已经有了一个名为 /usr/local/src/ssh-2.0.13的子目录，可以使用cd命令进入到其中：

```
[root@ford src] # cd /usr/local/src/ssh-2.0.13
```

4.2.2 查找软件包中的有关文档

进入到全部源代码所在的子目录之后，要从查找文档入手。一定要阅读随源代码而来的文档。如果有特殊的编译指导、注释或者警告，几乎都会在这里提到。首先阅读有关的文件

会让你避免不少的麻烦。

哪些是有关的文件呢？在发行版本中，通常会有两个文件：存放在源代码子目录第一级里面的README和INSTALL文件。README文件通常包括了软件包的说明、附加文档（包括安装文档）的索引、软件包作者的联系方法等等； INSTALL文件通常是编译和安装这个软件包的操作指导。

这些并不是绝对的，每个软件包都有需要谨慎对待的地方。最简单的检查方法就是列出子目录的内容清单，看看其中有没有比较明显的额外文档的标志。对 SSH来说，可以看到有一个名为SSH2.QUICKSTART的文件。从文件名上猜测，其中很可能有配置、编译和安装SSH2的操作步骤。有的软件包使用不同的大小写组合： readme、README、ReadMe等等。而有的介绍性文件还有这样的文件名： README.1ST或者README.NOW等等。

额外资料的另外一个存放位置可能会是名为“ doc ”或者“ documentation ”之类的子目录。对SSH来说，只有apps、include和lib三个子目录。

要想阅读一个文本文件，可以使用 more命令，如下所示：

```
[root@ford ssh-2.0.13] # more README
```

要想使用一个文本编辑器程序阅读文档，可以使用 pico命令，如下所示：

```
[root@ford ssh-2.0.13] # pico README
```

窍门 要想快速得到一份源代码所在目录中全部的子目录清单，可以使用如下的命令：

```
[root@ford ssh-2.0.13] # ls -l | grep drwx
```

4.2.3 配置新软件包

大多数软件包都会带有一个执行自动配置操作的脚本程序；除非它们的文档里有另外的说明。这些脚本程序通常都命名为“ configure ”，并且可以有参数。对各种不同的配置脚本程序来说，可以使用的参数实在太多了，但是每个软件包都有各自不同的特色。每个软件包都有大量的能够被激活或者禁止的功能，或者需要在编译时设置一些特殊的值，它们都需要进行配置。

要想查看软件包都有哪些配置选项，只需执行下面的命令：

```
[root@ford ssh-2.0.13] # ./configure --help
```

注意，在单词“ help ”前面有两个短划线（--）。

对SSH来说，我们将看的一个很长的清单，其中列出了许多种能够被激活或者禁止的功能（请阅读第17章中对SSH各项参数含义的说明）。

最常见的一个参数是 --prefix。这个参数允许读者设置软件包开始安装的基本子目录。缺省的情况下，大多数软件包都会安装到 /usr/local子目录的下级子目录里。举例来说， SSH的服务器组件叫做sshd，它就被安装到/usr/local/sbin子目录中。

把需要设置的参数都设置好以后，最后一次运行 configure将建立一个特殊的文件类型“ makefile ”（制作文件），制作文件是编译阶段的基石。

4.2.4 编译新软件包

编译软件包是一个很简单的操作。读者只需要运行 make命令，如下所示：

```
[root@ford ssh-2.0.13] # make
```

make工具程序将读入所有由configure脚本程序建立的制作文件。这些制作文件告诉 make哪些文件需要被编译以及按照怎样的顺序对它们进行编译 —— 这非常关键，因为可能会有上百个源程序文件。

根据计算机系统的速度、可用内存以及系统是否忙于处理其他事情，编译过程可能需要一定的时间才能完成，不必惊讶。

当make工作的时候，会在屏幕上显示出正在执行的每一个命令，以及与这个命令相关的全部参数。这些输出通常都是编译器的调用声明和所有传递给编译器的参数 —— 这个过程非常枯燥，甚至是程序员都愿意让它自动进行！

如果编译器顺利地完成了工作，就不会出现什么错误信息。大多数编译器的错误信息十分清楚和明确，因此不必担心可能会漏掉一个错误。如果确实看到有一错误，也不用慌张。大多数错误信息并不反映出程序本身出现了一个问题，通常都是系统这里或者那里的问题。典型情况下，这些信息大多是因为文件访问权限不正确而产生的（参见第6章中的chmod命令），或者是因为文件没有找到。在后一种情况，请检查自己的路径设置至少要包括 /bin、/sbin、/usr/bin、/usr/sbin、/usr/local/bin、/usr/local/sbin以及/usr/X11R6/bin等几个子目录。读者可以使用下面的命令查看自己的路径设置：

```
[root@ford ssh-2.0.13] # echo $PATH
```

请阅读第5章中关于setenv命令的介绍，这样就可以把路径设置正确了。

别慌，好好读一读错误信息。虽然它的格式可能有点怪，但是可以明白地说明到底是哪出了问题，你可以尽快地修复它。如果错误很绕人，可以去查查随软件包来的文档中是否有可以寻求帮助的邮件表或者 E-mail地址。大多数开发人员是很乐于提供帮助的，但是你需要注意礼貌，还要问到点儿上（换句话说，不要在电子邮件的开始写上一大段抱怨的话，问程序员为什么他们的软件这么糟糕！）

4.2.5 安装新软件包

与编译过程差不多，安装过程一般都会进展得比较顺利。大多数情况下，当编译过程结束之后，读者只需要执行下面的命令：

```
[root@ford ssh-2.0.13] # make install
```

这个命令将启动安装脚本程序（它经常嵌入在制作文件里）。因为make命令会在执行每一个命令的时候把它显示出来，所以读者将会看到许许多多的文字掠过眼前。不必操心这些文字 —— 它们都是些正常的东西。如果没有看到什么错误信息，就说明这个软件包安装好了。

如果确实看到了错误信息，通常也是因为访问权限的问题。检查出现问题时最后一个安装的文件，然后检查那里所有的访问权限要求，把需要的文件加上去。在这一步可能需要使用chmod、chown和chgrp命令；详细情况请阅读第6章。

4.2.6 安装完成后的清理工作

软件包安装好以后，需要再做一些清理工作，把安装过程建立的所有临时文件都删除掉。因为读者有原始的源代码tarball文件，所以把读者进行源代码编译工作处的那些子目录全部删除是安全的。对SSH软件来说，就是要删除 /usr/local/src/ssh-2.0.13子目录。先进入打算删除

的子目录的上一级目录。在本章的例子中，就是 usr/local/src子目录。

```
[root@ford ssh-2.0.13] # cd usr/local/src
```

现在使用rm命令删除那个子目录，如下所示：

```
[root@ford src] # rm -rf ssh-2.0.13
```

警告 rm命令，特别是它的-rf参数是很危险的。它将递归地删除某个子目录下的全部文件，而其过程中对“任何”文件都不会给出提示或者半途而废。如果作为一个系统管理员执行这个命令，它确实有可能对自己的系统造成潜在的威胁。一定要确定删除的确实是你想要删除的东西。没有什么“ undelete ”（反删除）命令。

4.3 小结

本章读者学习了使用 RPM和自行编译软件两种方法在 Linux操作系统中对软件进行安装。我们希望以上的内容可以消除读者对与源代码开放系统打交道时的恐惧！

需要记住的关键内容有以下几个方面：

- 使用RPM进行安装的典型命令是rpm -i packagename.rpm。
- 使用RPM进行升级的典型命令是rpm -U packagename.rpm。
- 使用RPM删除已安装软件的典型命令是rpm -e packagename.rpm。
- 大多数源代码都是“ tarball ”格式的文件，它们可以使用 tar命令进行解压缩。
- 解压缩完成后，查阅软件包附带的文档是非常重要的。
- 配置软件包可以使用./configure命令。
- 编译软件包可以使用make命令。
- 安装编译好的软件可以使用make install命令。