目录

[RPM部分 1](#_Toc322208927)

[简介 1](#_Toc322208928)

[安装 2](#_Toc322208929)

[卸载 2](#_Toc322208930)

[升级 2](#_Toc322208931)

[查询 2](#_Toc322208932)

[校验 3](#_Toc322208933)

[Rpm常用命令选项组合 3](#_Toc322208934)

[YUM部分 4](#_Toc322208935)

[简介 4](#_Toc322208936)

[安装 5](#_Toc322208937)

[升级 5](#_Toc322208938)

[卸载 5](#_Toc322208939)

[查询 5](#_Toc322208940)

[spec文件 6](#_Toc322208941)

[简介 6](#_Toc322208942)

[例子 6](#_Toc322208943)

[解释 7](#_Toc322208944)

## RPM部分

### 简介

RPM全称是Red Hat Package Manager（Red Hat包管理器）。

RPM安装软件包，通常要执行检查软件包的依赖（Dependency）、检查软件包的冲突（Conflicts）、解压软件包并存放到相应位置等操作。

RPM软件包分为二进制包（Binary）以及源代码包（Source）两种。二进制包可以直接安装在计算机中，而源代码包将会由RPM自动编译、安装。源代码包经常以src.rpm作为后缀名

Rpm在安装软件包的时候解决不了软件包的依赖性关系，而yum解决了此问题

在Redhat Linux中,其rpm包的命名规则为:

软件名-版本号-释出号.体系号.rpm

例： zsh-4.2.6-1.i386.rpm

zsh为软件名；

4.2.6为版本号 (4为主版本号、2为次版本号、6为修订版本号) ；

1为释出号（发布号）；

rpm为扩展名；

指令与语法如下：

rpm <选项>　<软件包名>

### 安装

* 安装选项
  + -i 安装一个以前从未安装过的软件包。
  + 例如：rpm –i zsh-4.2.6-1.i386.rpm
* 其它配用选项
  + -v 将安装过程打印在屏幕上；
  + -h 以hash字符“#”显示安装进度条；

rpm -ivh xxx.rpm 安装一个包

### 卸载

* 卸载选项
  + -e 卸载一个已经安装过的软件包。
  + 例如：rpm –e 文件名
* 注意
  + rpm对一个未安装的软件包进行操作，参数为软件包名。
  + rpm对一个已经安装过的软件包进行操作，参数为软件名。

rpm -e xxx.rpm 删除一个包

### 升级

* 升级选项
  + -U 对一个已经安装的旧版本软件包进行升级（如果以前没有安装过此包则安装它）。
  + -F 对一个已经安装的旧版本软件包进行升级。
* 其它配用选项

--force 强制执行，它与-Uvh组合使用可重新安装某个已经安装的软件包

对低版本软件进行升级,省去我们卸载后再安装新软件的麻烦，要升级某个软件，只须执行如下命令：rpm -uvh <文件名>，注意：此时的文件名必须是要升级软件的升级补丁

rpm -Uvh xxx.rpm 升级一个包

### 查询

* 通过RPM来查询的指令与语法如下：
  + rpm <选项>　<软件名｜软件包名｜文件名>
* 查询选项
  + –q 软件名：查询某一个RPM包是否已安装
  + –qi 软件名：查询某一个RPM包的详细信息
  + –ql 软件名：列出某RPM包中所包含的文件
  + –qf 文件名：查询某文件属于哪一个RPM包
  + –qa 软件名：列出当前系统所有已安装的包
  + –qip 软件包：列出未安装RPM包的详细信息
  + rpm -qilp xxx.rpm 列出一个未被安装进系统的RPM包文件中包含有哪些文件

【安装之前，一般都要先查看一下这个软件包里有什么内容，统将会列出这个软件包的详细资料，包括含有多少个文件、各文件名称、文件大小、创建时间、编译日期等信息】

* + –qlp 软件包：列出未安装RPM包中的文件

【在安装时不一定全部安装,查看软件包将会在系统里安装哪些部分，以方便我们的选择】

rpm -qpi xxx.rpm 显示包描述

rpm -qpl out.rpm 显示包的文件的列表

rpm -qgR out.rpm 显示包的依赖关系

rpm -q out.rpm 包是否已经安装

rpm -i --test out.rpm 安装测试

### 校验

* 当一个软件包被安装后，用户可以对其进行检验，以检测软件是否被用户修改过。
* 通过RPM来校验软件包的指令与语法如下：
  + rpm <选项>　<软件名>
* 校验选项
  + -V 校验软件；-Va 校验所有软件
* 校验出被修改地方
  + **S** 文件大小；**M** 文件权限与类型；**5** MD5求和；**U** 文件的所属用户；**G** 文件的所属组；**T** 更改时间

### Rpm常用命令选项组合

* rpm -ivh xxx.rpm 安装一个包，rpm包安装详细过程，显示进度
* rpm -Uvh xxx.rpm 升级一个包
* rpm --force 强制安装
* rpm -e xxx.rpm卸载，删除一个包 -evh
* rpm -q xxx.rpm 查询一个包是否被安装
* rpm -qi xxx.rpm 得到被安装的包的信息
* rpm -ql xxx.rpm 列出该包中有哪些文件，查看安装后文件列表
* rpm -qf files 查找文件所属的RPM包,快速判定某个文件属于哪个软件包
* rpm -qa 列出所有被安装的rpm package ，查看该软件是否安装
* rpm -qal 包 查看包中文件列表
* rpm -qilp xxx.rpm 列出一个未被安装进系统的RPM包文件中包含有哪些文件。
* rpm -qpi xxx.rpm 显示包描述
* rpm -qpl out.rpm 显示包的文件的列表
* rpm -qgR out.rpm 显示包的依赖关系
* rpm -q out.rpm   包是否已经安装
* rpm -i --test out.rpm 安装测试
* rpm -aid 自动安装依赖的包
* rpm –V out.rpm 校验rpm软件包是否有问题
* rpm –Vp校验rpm软件包是否有问题
* rpm –Va 校验所有的rpm软件包，列出所有损坏的文件，以便进行修复

## YUM部分

### 简介

* yum是Yellow dog Updater Modified的简称
* yum的关键之处是要有可靠的repository，顾名思义，这是软件的仓库，它可以通过http或ftp共享出来，也可以是本地软件池。
* 仓库中必须包含rpm的header， header包括了rpm包的各种信息，包括描述，功能，提供的文件，依赖性等。
* 客户端在安装的时候便下载header并加以分析，才能自动地从服务端下载软件并一条龙地完成安装任务。
* yum客户端的配置文件放在本地的/etc/yum.repos.d/\*.repo
* 配置文件的格式为
  + [仓库名1]

name=仓库描述

baseurl=仓库位置

enabled=0/1(0：不激活配置；1：激活此配置)

gpgcheck=0/1(0:安装前不对RPM包检测1:安装前对RPM包检测)

gpgkey=GPG文件的位置

* + [BASE]  
    name=Red Hat Enterprise Linux Server  
    baseurl=ftp://ip/pub/Server  
    enabled=1  
    gpgcheck=1  
    gpgkey=file:/// rpm-gpg-path/RPM-GPG-KEY-redhat-release

### 安装

* + 语法：yum 　<选项>　<软件包|软件名|组名>
* 安装选项
  + install package 安装指定的安装包package
  + localinstall package 本地安装软件包
  + groupinstall package 组安装
* 补充说明
  + install是客户端从远程仓库中下载软件并安装
  + localinstall 是安装一个本地已经下载来的的软件包。
  + groupinstall 如果仓库为软件包分了组，则可以通过安装此组来完成安装这个组里面的所有软件包。

### 升级

* + 语法：yum　 <选项>　[软件包|软件名|组名]
* 升级选项
  + update 更新全部可以更新的软件包
  + update package 更新指定程序包package
  + check-update 检查系统中可以被升级的软件包。
  + groupupdate group1 升级组group1 里面的软件包。

### 卸载

* + 语法：yum　 <选项>　[软件包|软件名|组名]
* 卸载选项
  + remove package1 ： 卸载软件包package1
  + erase　 package1 ： 卸载软件包package1
  + groupremove group1 ：卸载组group1里面所包括的软件包。

### 查询

* + 语法：yum　 <选项>　[软件包|软件名|组名]
* 查询选项
  + info <package> ：显示安装包信息
  + list ：显示所有已经安装和仓库中可以安装的程序包
  + list <package> ：显示指定软件包安装情况
  + grouplist ：显示仓库中有哪些组
  + deplist <package1> ：查看程序package1依赖情况
  + groupinfo <group1> ：显示程序组group1信息
  + search <keyword>：查找软件包

清除缓存

yum clean all

## spec文件

### 简介

要想制作一个RPM格式的软件包,需要编写软件包描述文件spec

其标准命名格式为:软件名-版本号-释出号.spec，这个文件,详细描述了有关该软件包的诸 多信息,如软件名,版本,类别,说明摘要,创建时要执行什么指令,安装时要执行什么操作,以及软件包所要包含的文件等等

### 例子

|  |
| --- |
| .spec文件（httpd.spec）  define contentdir /var/www  %define suexec\_caller apache  %define mmn 20051115    %ifarch ia64  # disable debuginfo on IA64  %define debug\_package %{nil}  %endif    Summary: Apache HTTP Server  Name: httpd  Version: 2.2.11  Release: 1  URL: http://httpd.apache.org/  Vendor: Apache Software Foundation  Source0: http://www.apache.org/dist/httpd/httpd-%{version}.tar.gz  License: Apache License, Version 2.0  Group: System Environment/Daemons  BuildRoot: %{\_tmppath}/%{name}-root  BuildPrereq: apr-devel, apr-util-devel, openldap-devel, db4-devel, expat-devel, findutils, perl, pkgconfig  BuildPrereq: /usr/bin/apr-1-config, /usr/bin/apu-1-config  Requires: apr >= 1.2.0, apr-util >= 1.2.0, gawk, /usr/share/magic.mime, /usr/bin/find, openldap  Prereq: /sbin/chkconfig, /bin/mktemp, /bin/rm, /bin/mv  Prereq: sh-utils, textutils, /usr/sbin/useradd  Provides: webserver  Provides: httpd-mmn = %{mmn}  Conflicts: thttpd  Obsoletes: apache, secureweb, mod\_dav    %description  Apache is a powerful, full-featured, efficient, and freely-available  Web server. Apache is also the most popular Web server on the  Internet.    %package devel  Group: Development/Libraries  Summary: Development tools for the Apache HTTP server.  Obsoletes: secureweb-devel, apache-devel  Requires: libtool, httpd = %{version}  Requires: apr-devel >= 1.2.0, apr-util-devel >= 1.2.0    %description devel  The httpd-devel package contains the APXS binary and other files  that you need to build Dynamic Shared Objects (DSOs) for Apache |

### 解释

Spec-关键字

|  |  |
| --- | --- |
| Name | 此域定义软件名 |
| Version | 此域定义版本号。仅当软件较以前有较大改变时才增加版本号。注: 版本号中不能含减号(-)字符 |
| Release | 此域定义释出号。若软件较以前改变较小,则仅增加释出号,不改变版本号。注: 释出号中亦不能含减号(-)字符  RPM利用上述的Name(软件名),Version(版本号),Release(释出号)及体系号来命名软件包,如本例输出的包裹文件名为lze-6.0-2.i386.rpm |
| Summary | 此域定义软件包简介,为一句话说明 |
| Group | 此域定义软件所属类别,详见<<精通RPM之五--查询篇>>,本例的Applications/Editors表示本软件属“应用/编辑器”类 |
| License | 此域定义软件适用的许可证或版权规则。该域也可用Copyright(版权)来定义,二者同意。许可证具体有: GPL(通用公共许可证,自由软件适用),BSD,MIT,Public,  Domain(公共域),Distributable(贡献),Commercial(商),Share(共享)等 |
| **Provides** | 此域定义软件包提供的功能,可重复多行。其描述格式为: Provides : 功能1 [,功能2] ... |
| **Requires** | 此域定义软件包所需的功能,可重复多行。其描述格式为:  　　Requires : 功能1 [比较符1 [序列号1:]版本号1[-释出号1]] [,功能2 [比较符2 [序列号2:]版本号2[-释出号2]]] ...  　　其中: \* []所括为可选项; 　　\* 比较符可使用<(小于),>(大于),=(等于),>=(大于等于)或<=(小于等于); 　　\* 序列号不选时,RPM默认为0; 　　\* 功能之间的逗号可选,也可使用空格进行分隔。 　　例子:Requires: aaa, bbb >= 3.0, ccc < 2:5.0-1 |
| **Conflicts** | 此域定义有哪些功能与本软件包相冲突(不能共存)。此域亦可在描述文件中书写多次。其描述格式形同Requires域,为:  　　Conflicts : 功能1 [比较符1 [序列号1:]版本号1[-释出号1]] [,功能2 [比较符2 [序列号2:]版本号2[-释出号2]]] ...  　　其中: \* []所括为可选项; 　　\* 比较符可使用<(小于),>(大于),=(等于),>=(大于等于)或<=(小于等于); 　　\* 序列号不选时,RPM默认为0; 　　\* 功能之间的逗号可选,也可使用空格进行分隔。 　　举个例子:  　　Conflicts : xxx=1:2.0 yyy>=3.0 |
| **BuildRoot** | 这个是安装或编译时使用的“虚拟目录”，考虑到多用户的环境，一般定义为： %{\_tmppath}/%{name}-%{version}-%{release}-root 或 %{\_tmppath}/%{name}-%{version}-%{release}-buildroot-%(%{\_\_id\_u} -n} 该参数非常重要，因为在生成rpm的过程中，执行make install时就会把软件安装到上述的路径中，在打包的时候，同样依赖“虚拟目录”为“根目录”进行操作。后面可使用$RPM\_BUILD\_ROOT 方式引用 |
| **Prefix** | %{\_prefix} 这个主要是为了解决今后安装rpm包时，并不一定把软件安装到rpm中打包的目录的情况。这样，必须在这里定义该标识，并在编写%install脚本的时候引用，才能实现rpm安装时重新指定位置的功能。  %{\_sysconfdir} 这个原因和上面的一样，但由于%{\_prefix}指/usr，而对于其他的文件，例如/etc下的配置文件，则需要用%{\_sysconfdir}标识 |
| **Build Arch** | 指编译的目标处理器架构，noarch标识不指定，但通常都是以/usr/lib/rpm/marcros中的内容为默认值 |
| **%prep** | 预处理脚本  **%setup -n %{name}-%{version}** 把源码包解压并放好  %setup 不加任何选项，仅将软件包打开。   %setup -n newdir 将软件包解压在newdir目录。   %setup -c 解压缩之前先产生目录。   %setup -b num 将第num个source文件解压缩。   %setup -T 不使用default的解压缩操作。   %setup -T -b 0 将第0个源代码文件解压缩。   %setup -c -n newdir 指定目录名称newdir，并在此目录产生rpm套件。  %setup -q 禁止输出信息 |
| **%build** | 开始构建包  在/usr/src/redhat/BUILD/%{name}-%{version}目录中进行make的工作 ，常见写法：  make |
| **%install** | 开始把软件安装到虚拟的根目录中  在/usr/src/redhat/BUILD/%{name}-%{version}目录中进行make install的操作。常见内容有： %makeinstall 是rpm定义的标准宏命令。也可以使用非标准写法：  make DESTDIR=$RPM\_BUILD\_ROOT install  或  make prefix=$RPM\_BUILD\_ROOT install  还可以使用常规的系统命令：  install -d $RPM\_BUILD\_ROOT/  cp -a \* $RPM\_BUILD\_ROOT/ |
| **%clean** | 清理临时文件 通常内容为：  [ "$RPM\_BUILD\_ROOT" != "/" ] && rm -rf "$RPM\_BUILD\_ROOT" rm -rf $RPM\_BUILD\_DIR/%{name}-%{version} |
| **其他** | * **%pre** rpm安装前执行的脚本 * **%post** rpm安装后执行的脚本 * **%preun** rpm卸载前执行的脚本 * **%postun** rpm卸载后执行的脚本 * **%files** 定义那些文件或目录会放入rpm中 **这里会在虚拟根目录下进行，千万不要写绝对路径，而应用宏或变量表示相对路径。** 如果描述为目录，表示目录中除%exclude外的所有文件。 * **%defattr (-,root,root)** 指定包装文件的属性，分别是(mode,owner,group)，-表示默认值，对文本文件是0644，可执行文件是0755 **%exclude** 列出不想打包到rpm中的文件 |

# 不安装rpm包提取rpm软件包中的文件

解决这个问题，方法是用rpm2cpio将rpm文件转换成cpio格式，然后再从cpio中提取文件，下面是详细命令：

rpm2cpio foo.rpm | cpio -idmv

rpm2cpio可以将rpm包转换成cpio格式

cpio命令中的-i是提取文件，-d表示根据rpm包中文件原来的路径建立目, -m是保持文件的更新时间, -v是显示处理的文件列表。