链滴

Linux/Unix 下的程序安装

作者：liumapp

原文链接：https://ld246.com/article/1496893843920

来源网站：链滴

<blockquote>
个人对 Linux 程序安装的理解
</blockquote>
<h2 id="软件包的分类">软件包的分类</h2>
<h3 id="源码包">源码包</h3>
<h5 id="优点">优点</h5>

开源，如果有足够的 C 语言编程能力，可以修改源代码；
可以自由选择所需要的功能
软件是编译安装，所以更加合适自己的系统，更加稳定也更加高效
卸载方便。（直接卸载安装目录即可，具体请参考下文）

<h5 id="缺点">缺点</h5>

安装过程步骤较多，尤其安装大型的软件集合时（如 LAMP 环境的搭建），容易出现拼写错误</l
>
编译时间比二进制包的安装要长
编译过程中一旦报错，新手很难解决问题，所以源码包安装并不适合初学者

<h3 id="二进制包-rpm包-系统默认包等等-下文我们统一视为rpm包-">二进制包(rpm 包、系统默包等等，下文我们统一视为 rpm 包)</h3>
<h5 id="优点-">优点</h5>

包管理系统简单，只通过几个简单的命令就可以实现包的安装、升级、查询和卸载
安装速度比源码包安装快得多

<h5 id="缺点-">缺点</h5>

已经编译好的包，所以不能看到源代码
功能选择不如源码包灵活
依赖性(用过 maven 或者 composer 的小伙伴应该清楚，当我装软件包 A，如果 A 要用到软件包 B，B 又要用到软件包 C，那我就必须先装 C，再装 B，最后再装 A)

<h6 id="依赖性详解">依赖性详解</h6>

树形依赖:a->b->c
环形依赖:a->b->c->a
模块依赖:a->b，而 b 是一个 C 的函数库，以 so 的后缀结尾，它具体在哪个包下面就要去网上面查了： <a href="https://ld246.com/forward?goto=http%3A%2F%2Fwww.rpmfind.net" tar
et="_blank" rel="nofollow ugc">模块依赖查询站点

<h3 id="脚本安装包">脚本安装包</h3>
通过 shell 脚本，把复杂的软件包安装过程写成了程序脚本，初学者可以执行程序脚本实现一键装。但实际安装的还是源码包和 rpm 包。
大家可以参考我的一个 lnmp 环境自动搭建的脚本包，源代码如下： <a href="https:/
ld246.com/forward?goto=https%3A%2F%2Fgithub.com%2Fliumapp%2Flnmp" target="_blan
" rel="nofollow ugc">lnmp
<h5 id="优点--">优点</h5>
安装简单、快捷
<h5 id="缺点--">缺点</h5>
完全丧失了自定义性。（不过这也算不上缺点，因为脚本本身也是开源的，只要你有 shell 编程能力，也可以进行自定义安装）
<h2 id="软件包的命名规则">软件包的命名规则</h2>
<h3 id="rpm包">rpm 包</h3>

我们以下面这个为例："httpd-2.2.15-15.el6.centos.1.i686.rpm"。

httpd 表示软件包名称
2.2.15 表示软件版本
15 表示软件的发布次数
el6.centos 表示适合的 Linux 平台
i686 表示适合的硬件平台
rpm 表示包的扩展名

<h2 id="安装位置">安装位置</h2>
<h2 id="rpm">rpm</h2>
如果是使用 rpm 包进行安装的话，虽然 rpm 可以指定安装位置，但我们一般还是不要指定位置直接安装即可。因为 rpm 包默认安装后，系统可以；对它的启动、删除进行快捷的管理。
<h5 id="默认位置">默认位置</h5>

/etc/ 配置文件安装目录
/usr/bin/ 可执行命令的安装目录
/usr/lib/ 程序所使用的函数库保存位置
/usr/share/doc/ 基本的软件使用手册保存位置
/usr/share/man/ 帮助文件保存位置

<h2 id="源码包-">源码包</h2>
安装的时候，一般位于“/usr/local/软件名”下面。因为通过源码进行安装，不向通过 rpm 安一样有卸载命令，它是没有卸载命令的，所以统一安装在指定的目录下。一旦我们没有指定安装目录那么系统将会按照 rpm 的流程来进行安装（安装在系统的“各个目录”），且没有卸载命令，所以除的时候就头疼着吧。
<h4 id="指定位置">指定位置</h4>
那么源码安装的时候，我们如何去指定它的安装位置呢？
很简单，一般源码包下面都会有类似于 configure 的文件。
我们可以通过输入以下命令
<pre><code class="highlight-chroma">./configure -help
</code></pre>
来查看这个命令到底有什么用，在打印出来的列表中可以看到一个 prefix 命令就是用来指定安位置的。
比如，要指定安装位置为/usr/local/project，那就是
<pre><code class="highlight-chroma">./configure --prefix=/usr/local/project
</code></pre>
注意，这个等于号之间不能有空格。
<h2 id="rpm跟源码包的区别">rpm 跟源码包的区别</h2>

rpm 是闭源项目，是已经编译好的程序，而源码包则是开源程序，在安装之前要先进行编译。

rpm 安装的服务可以使用系统服务管理命令（service）来管理，例如 rpm 安装的 apache 的启方法是：
<pre><code class="highlight-chroma"> /etc/rc.d/init.d/httpd start
  或者
  service httpd star</pre>

</code></pre>
而源码包安装的服务则不能被服务管理命令管理，因为没有安装到默认的路径中。所以只能使用对路径进行管理。如：
<pre><code class="highlight-chroma"> /usr/local/apache2/bin/apachectl start
</code></pre>
但我们也有取巧的办法，来让 service 启动：把 apachectl 这个文件拷贝到/etc/rc.d/init.d 目录，就可以了。

rpm 因为是厂商提前编译好的，所以在执行效率上，会比在本机编译的源码包低。

<h2 id="安装前提">安装前提</h2>
<h2 id="rpm-">rpm</h2>
首先我们先理解一下 rpm 包的“包全名”与“包名”这两个概念

包全名
操作的包是没有安装的软件包时，使用包全名。而且要注意路径。

包名
操作已经安装的软件包时，使用包名（一般就是它的项目名），是搜索/var/lib/rpm/中的数据。

<h5 id="rpm包的校验">rpm 包的校验</h5>
<pre><code class="highlight-chroma">rpm -V 已安装的包名
</code></pre>

-V 校验指定的 rpm 包中的文件（verify）

如果上述命令执行后，没有返回信息出现，那说明一切正常，如果有，那么就说明它要搞事情了

S 文件大小是否改变
M 文件的类型或者文件的权限(rwx)是否被改变
5 文件 MD5 校验和是否被改变（可以理解为文件内容是否改变）
D 设备的主从代码是否改变
L 文件路径是否改变
U 文件的所有者是否改变
G 文件的属组是否改变
T 文件的修改时间是否改变

这里补充一下文件类型的知识，因为返回信息的最后，会出现一个当前文件的类型：

c 配置文件（config file）
d 普通文档（documentation）
g "鬼"文件（ghost file），很少见，就是该文件不应该被这个 RPM 所包含
L 授权文件（license file）
r 描述文件（read me）

<h5 id="rpm包中文件的提取">rpm 包中文件的提取</h5>
<pre><code class="highlight-chroma">rpm2cpio 包全名 | cpio -idv .文件绝对路径
</code></pre>

rpm2cpio 将 rpm 包转换为 cpio 格式的命令
cpio 是一个标准工具，它用于创建软件档案文件和从档案文件中提取文件

-i copy-in 模式，还原
-d 还原时自动新建目录
-v 显示还原过程

<h2 id="源码包--">源码包</h2>

要有 C 编译器
我们可以通过以下的命令来检查系统有没有 C 编译器：
<pre><code class="highlight-chroma"> rpm -qa | grep gcc
</code></pre>
当然，如果我们使用类 Unix 的系统，比如说 Mac OS，我自己的电脑用的就是，这种情况下，打上面的命令是不会有结果的，因为 Unix 本身就可以编译 C 语言，不需要额外的服务。

下载到程序的源码包，将其保存在/usr/local/src 目录下，这个目录保存程序源码。安装目录则它的上一级目录下。

<h2 id="安装-升级与卸载">安装、升级与卸载</h2>
<h2 id="rpm--">rpm</h2>
<h5 id="安装命令">安装命令</h5>
<pre><code class="highlight-chroma">rpm -ivh 包全名
</code></pre>

-i(install) 安装
-v(verbose) 显示详细信息
-h(hash) 显示进度
--nodeps 不检测依赖性（这个命令在正式服务器环境上就没有用过，也不要去用）

<h5 id="查询命令">查询命令</h5>
我们手动的安装 rpm 包才可以通过查询命令查询到，如果通过 yum 命令进行安装，那么就不能使用查询命令。

查询
<pre><code class="highlight-chroma"> rpm -q 包名（查询包是否安装,-q query）

  rpm -qa (查询所已经安装的rpm包，-a all)

原文链接：Linux/Unix 下的程序安装

  rpm -qi 包名（
时查询包安装的详细信息，比如安装时间等等）

  rpm -qip 包全名
查询还没有安装的包信息）

  rpm -ql 包名 (-l
询包中各个文件的安装位置）

  rpm -qf 系统某
个文件名(-f 查询系统文件属于哪个软件包）

  rpm -qR 包名(-R
查询软件包的依赖性，但这个命令其实意义不大，因为我们需要的只是缺少什么依赖）
</code></pre>
rpm -qa 一般配合管道符“|”来使用，比如
<pre><code class="highlight-chroma"><span class="highlight
cl"> rpm -qa | grep http 就表示去查询包含http关键字的所有已安装的包名
</code></pre>

<h5 id="升级命令">升级命令</h5>
<pre><code class="highlight-chroma"><span class="highlight
cl">rpm -Uvh 包全名(比当前版本更高的包，如果没有下载过，那么这个命令就是纯粹的安装命令，
果版本没有旧包的高，那么将不会进行安装）
</code></pre>

-U(upgrade) 升级

<h5 id="卸载命令">卸载命令</h5>
在卸载之前，如果包有自己的依赖，那么必须先把依赖的包进行删除（同样的命令），再来删除
卸载的包
<pre><code class="highlight-chroma"><span class="highlight
cl">rpm -e 包名
</code></pre>

-e(erase) 卸载

<h2 id="yum">yum</h2>
将所有软件包放到官方服务器上，当进行 yum 在线安装时，可以自动解决依赖性问题。
<h5 id="插曲">插曲</h5>
redhat 的 yum 服务是收费的，CentOS 的 yum 服务是免费的，所以本人接手过的 10 几台服
器清一色的 CentOS。
<h5 id="yum也有配置文件噢">yum 也有配置文件噢</h5>
这个配置文件位于：/etc/yum.repos.d/CentOS-Base.repo。
下面，对它的一些参数做一个备注

[base] 容器名称，一定要放在[]中
name 容器说明，可以自己随便写
mirrorlist 镜像站点，这个可以注释掉(因为 mirrolist 和 baseurl 两者保留一个即可)
baseurl 我们的 yum 源服务器的地址。默认是 CentOS 官方的 yum 源服务器，是可以使用的，
果觉得慢可以改成自己喜欢的 yum 源地址
enabled 此容器是否生效，如果不写或者写成 enable＝1 都是生效的意思，写成 enable＝0 就
不生效

gpgcheck 如果是 1 是指 rpm 的数字证书生效，如果是 0 则不生效
gpgkey 数字证书的公钥文件保存位置。不用修改

<h5 id="查询命令-">查询命令</h5>
<pre><code class="highlight-chroma"><span class="highlight
cl">yum list
#查询所有可用软
包列表

yum search 关键字
#搜索服务器上所
和关键字相关的包
</code></pre>
<h5 id="安装命令-">安装命令</h5>
<pre><code class="highlight-chroma"><span class="highlight
cl">yum -y install 包名
</code></pre>

-y 自动回答 yes
install 下载

<h5 id="升级命令-">升级命令</h5>
<pre><code class="highlight-chroma"><span class="highlight
cl">yum -y update 包名
</code></pre>
但是请注意，这是一条非常危险的命令，因为一旦我们没有输入包名，只打了一个 yum -y upda
e ，那就意味着升级 Linux 下面所有的程序...包括我们的 Linux 内核
更新 Linux 内核意味着什么？
意味着我们之前对服务器内核做的所有配置全部狗带，新的服务器内核啥配置也没做(Linux 新的
核必须人为配置内核后才能够启动)，如果我们的是远程服务器，比如我常用的阿里云 ECS，那基本
味着系统瘫痪了....
所以对于 CentOS6.3 及以前的版本，一条简单的 yum -y update 就能够直接瘫痪系统。
理由：
就像我们前文说到的，安装软件有它的依赖性，那么卸载软件同样有它的依赖性。而我们在卸载
件的时候，并不能保证它有没有被系统的软件所依赖。
<h5 id="卸载命令-">卸载命令</h5>
<pre><code class="highlight-chroma"><span class="highlight
cl">yum -y remove 包名
</code></pre>
但尽量不要去使用卸载命令
因为管理服务器应该有以下的原则：
服务器使用最小化安装，用什么软件安装什么，尽量不卸载。
<h5 id="软件组管理命令">软件组管理命令</h5>
<pre><code class="highlight-chroma"><span class="highlight
cl">yum grouplist
列出所有可用的
件组列表

yum groupinstall
软件组名
安装指定软件组
组名可以由grouplist查询出来

yum groupremove

软件组名
卸载指定软件组
</code></pre>
<h2 id="源码包---">源码包</h2>

解压源码包并进入源码包目录

通过./configure 进行软件的配置与检查:(一般而言，每个源码包下面都会一个 INSTALL 或者 README 文件，说明如何进行 configure)

定义需要的功能选项，具体的选项可以通过./configure --help 来查看
检查系统环境是否符合安装要求
把定义好的功能选项和检测系统环境的信息都写入 Makefile 文件，用于后续的编辑

通过 make, make install 进行编译。
如果在 make 的过程中，有 error 等错误信息产生，那么先执行 make clean 再去检查错误原因，如是 make install 过程报错了，那就只能删除整个目录从头再来。

