[Linux软件安装管理之——dpkg与apt-\*详解](https://segmentfault.com/a/1190000011463440)

[https://avatar-static.segmentfault.com/335/109/3351090335-569f0f26a2a86_small](https://segmentfault.com/t/ubuntu/blogs)

一、序言

上一篇文章《[Linux软件安装管理之——源码安装详解](https://segmentfault.com/a/1190000011200004?share_user=1030000007255638)》详细介绍了Linux平台下的源码包安装原理，虽然使用源代码进行软件编译可以具有定制化的设置，但对于Linux distribution的发行商来说，则有软件管理不晚的问题，毕竟不是每个人都会进行源代码编译的。这个问题将会严重的影响linux平台上软件的发行与推广。

为了解决上述的问题，厂商先在他们的系统上面编译好了我们用户所需要的软件，然后将这个编译好并可执行的软件直接发布给用户安装。不同的 Linux 发行版使用不同的打包系统，一般而言，大多数发行版分别属于两大包管理技术阵营： Debian 的”.deb”，和 Red Hat的”.rpm”。也有一些重要的例外，比方说 Gentoo， Slackware，和 Foresight，但大多数会使用这两个基本系统中的一个。

这里将要介绍的是Debian系列发行版的dpkg与apt-\*的详细使用方法，如需要了解Red Hat系列发行版的包管理系统可查看我的另一篇文章：《[Linux软件安装管理之——RPM与YUM详解](https://segmentfault.com/a/1190000011200461?share_user=1030000007255638)》。

注：本人系统版本【Ubuntu 12.04 LTS】

二、dpkg

1、简介

dpkg - package manager for Debian （debian系统的包管理工具）。

dpkg is a tool to install, build, remove and manage Debian packages，dpkg是Debian的一个底层包管理工具，主要用于对已下载到本地和已安装的软件包进行管理。

dpkg这个机制最早由Debian Linux社区所开发出来的，通过dpkg的机制，Debian提供的软件就能够简单的安装起来，同时能提供安装后的软件信息，实在非常不错。只要派生于Debian的其它Linux distributions大多使用dpkg这个机制来管理，包括B2D，Ubuntu等。

2、deb软件包名规则

格式为：Package\_Version-Build\_Architecture.deb

如：nano\_1.3.10-2\_i386.deb

* 软件包名称(Package Name): nano
* 版本(Version Number):1.3.10
* 修订号(Build Number):2
* 平台(Architecture):i386

3、dpkg软件包相关文件介绍

/etc/dpkg/dpkg.cfg dpkg包管理软件的配置文件【Configuration file with default options】

/var/log/dpkg.log dpkg包管理软件的日志文件【Default log file (see /etc/dpkg/dpkg.cfg(5) and option --log)】

/var/lib/dpkg/available 存放系统所有安装过的软件包信息【List of available packages.】

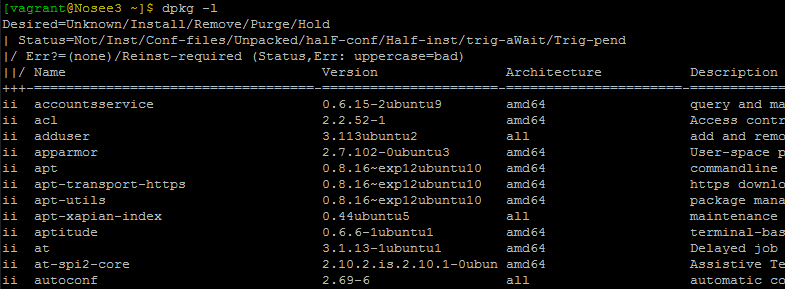
/var/lib/dpkg/status 存放系统现在所有安装软件的状态信息

/var/lib/dpkg/info 记安装软件包控制目录的控制信息文件

4、dpkg数据库

dpkg 使用文本文件作为数据库来维护系统中软件，包括文件清单, 依赖关系, 软件状态, 等等详细的内容,通常在 /var/lib/dpkg 目录下。 通常在 status 文件中存储软件状态和控制信息。 在 info/ 目录下备份控制文件， 并在其下的 .list 文件中记录安装文件清单， 其下的 .mdasums 保存文件的 MD5 编码。

例：查询dpkg数据库（显示所有已安装的Deb包）



如上图，每条记录对应一个软件包，注意每条记录的第一、二、三个字符，这就是软件包的状态标识，后边依此是软件包名称、版本号和简单描述。

1）第一字符为期望值(Desired=Unknown/Install/Remove/Purge/Hold)，它包括：

u Unknown状态未知,这意味着软件包未安装,并且用户也未发出安装请求.

i Install用户请求安装软件包.

r Remove用户请求卸载软件包.

p Purge用户请求清除软件包.

h Hold用户请求保持软件包版本锁定.

2）第二列,是软件包的当前状态(Status=Not/Inst/Conf-files/Unpacked/halF-conf/Half-inst/trig-aWait/Trig-pend)

n Not软件包未安装.

i Inst软件包安装并完成配置.

c Conf-files软件包以前安装过,现在删除了,但是它的配置文件还留在系统中.

u Unpacked软件包被解包,但还未配置.

f halF-conf试图配置软件包,但是失败了.

h Half-inst软件包安装,但是没有成功.

w trig-aWait触发器等待

t Trig-pend触发器未决

3）第三列标识错误状态,第一种状态标识没有问题,为空. 其它符号则标识相应问题（Err?=(none)/Reinst-required (Status,Err: uppercase=bad)）

h 软件包被强制保持,因为有其它软件包依赖需求,无法升级.

r Reinst-required，软件包被破坏,可能需要重新安装才能正常使用(包括删除).

x 软包件被破坏,并且被强制保持.

案例说明：

ii —— 表示系统正常安装了该软件

pn —— 表示安装了该软件，后来又清除了

un —— 表示从未安装过该软件

iu —— 表示安装了该软件，但是未配置

rc —— 该软件已被删除，但配置文件仍在

5、dpkg子命令

为了方便用户使用，dpkg不仅提供了大量的参数选项, 同时也提供了许多子命令。

比如：

dpkg-deb、dpkg-divert、dpkg-query、dpkg-split、dpkg-statoverride、start-stop-daemon

这里将不会对dpkg的命令作详细说明，如需了解可自行查阅资料。

三、dpkg使用手册

1、安装

**1）安装相关命令**

dpkg -i package-name.deb # --install,  
安装软件包，必须是deb包的完整名称。（软件的安装可被拆分为两个对立的过程“解包”和“配置”）

dpkg --unpack package-name.deb #  
“解包”：解开软件包到系统目录但不配置,如果和-R一起使用，参数可以是一个目录

dpkg --configure package-name.deb #“配置”：配置软件包

dpkg -c package-name.deb #列出 deb 包的内容

**2）安装相关选项**

-R, --recursive Recursively handle all regular files matching pattern \*.deb found at specified directories and all of its

1. This can be used with -i, -A, --install, --unpack
2. --avail actions（递归地指向特定目录的所有安装包，可以结合-i, -A, --install, --unpack 与

--avail一起使用）

2、移除软件包

dpkg -r package-name # --remove， 移除软件包，但保留其配置文件

dpkg -P package-name # --purge， 清除软件包的所有文件（removes everything,  
including conffiles）

3、查询

dpkg -l package-name-pattern # --list, 查看系统中软件包名符合pattern模式的软件包

dpkg -L package-name # --listfiles, 查看package-name对应的软件包安装的文件及目录

dpkg -p package-name # --print-avail, 显示包的具体信息

dpkg -s package-name # --status, 查看package-name（已安装）对应的软件包信息

dpkg -S filename-search-pattern # --search,  
从已经安装的软件包中查找包含filename的软件包名称 （Tip：也可使用子命令dpkg-query来进行查询操作）

例1：列出系统上安装的与dpkg相关的软件包

dpkg -l \**\*dpkg\***

例2：查看dpkg软件包安装到系统中的文件

dpkg -L dpkg

更多dpkg的使用方法可在命令行里使用man dpkg来查阅 或直接使用dpkg --help。

三、apt

1、apt简介

虽然我们在使用dpkg时，已经解决掉了 软件安装过程中的大量问题，但是当依赖关系不满足时，仍然需要手动解决，而apt这个工具解决了这样的问题，linux distribution 先将软件放置到对应的服务器中，然后分析软件的依赖关系，并且记录下来，然后当客户端有安装软件需求时，通过清单列表与本地的dpkg以存在的软件数据相比较，就能从网络端获取所有需要的具有依赖属性的软件了。

2、工作原理

Ubuntu采用集中式的软件仓库机制，将各式各样的软件包分门别类地存放在软件仓库中，进行有效地组织和管理。然后，将软件仓库置于许许多多的镜像服务器中，并保持基本一致。这样，所有的Ubuntu用户随时都能获得最新版本的安装软件包。因此，对于用户，这些镜像服务器就是他们的软件源（Reposity）

然而，由于每位用户所处的网络环境不同，不可能随意地访问各镜像站点。为了能够有选择地访问，在Ubuntu系统中，使用软件源配置文件/etc/apt/sources.list列出最合适访问的镜像站点地址。

例1：apt-get的更新过程

1）执行apt-get update

2）程序分析/etc/apt/sources.list

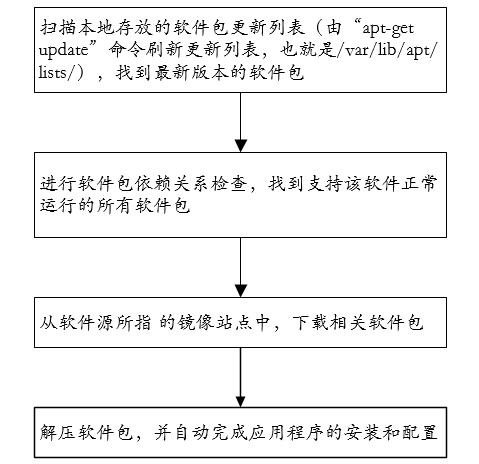
3）自动连网寻找list中对应的Packages/Sources/Release列表文件，如果有更新则下载之，存入/var/**lib**/**apt**/**lists**/目录

4）然后 apt-get install 相应的包 ，下载并安装。

即使这样，软件源配置文件只是告知Ubuntu系统可以访问的镜像站点地址，但那些镜像站点具体都拥有什么软件资源并不清楚。若每安装一个软件包，就在服务器上寻找一遍，效率是很低的。因而，就有必要为这些软件资源列个清单（建立索引文件），以便本地主机查询。

用户可以使用“apt-get update”命令刷新软件源，建立更新软件包列表。在Ubuntu Linux中，“apt-get update”命令会扫描每一个软件源服务器，并为该服务器所具有软件包资源建立索引文件，存放在本地的/var/lib/apt/lists/目录中。 使用apt-get执行安装、更新操作时，都将依据这些索引文件，向软件源服务器申请资源。因此，在计算机设备空闲时，经常使用“apt-get update”命令刷新软件源，是一个好的习惯。

例2：apt-get install原理图



3、apt相关文件

var/lib/dpkg/available 文件的内容是软件包的描述信息, 该软件包括当前系统所使用的Debian 安装源中的所有软件包,其中包括当前系统中已安装的和未安装的软件包.

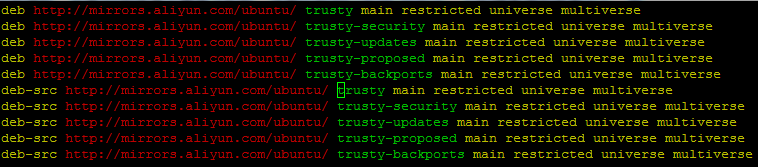
/etc/apt/sources.list 记录软件源的地址（当你执行 sudo apt-get install xxx 时，Ubuntu 就去这些站点下载软件包到本地并执行安装）

/var/cache/apt/archives 已经下载到的软件包都放在这里（用 apt-get install 安装软件时，软件包的临时存放路径）

/var/lib/apt/lists 使用apt-get update命令会从/etc/apt/sources.list中下载软件列表，并保存到该目录

4、源文件

apt的源文件由配置文件/etc/apt/sources.list指定，该文件配置内存如下图所示：



镜像服务器的存放位置如：



通过解压Packages.gz后我们可以看到所有软件包的基本信息，并且Packages.gz 已经包含所有所依赖的软件及其版本信息、以及下载地址。

四、apt使用手册

1、apt-get

apt-get is the command-line tool for handling packages, and may be considered the user's "back-end" to other tools using the APT library. Several "front-end" interfaces exist, such as dselect(1), aptitude(8), synaptic(8) and wajig(1).

apt-get 是一个下载安装软件包的简单命令行接口。最常用的命令是update(更新)和install(安装)。

用法： apt-get [选项] 命令

常用选项：

-h 本帮助文件。

-q 输出到日志 - 无进展指示

-qq 不输出信息，错误除外

-d 仅下载 - 不安装或解压归档文件

-s 不实际安装。模拟执行命令

-y 假定对所有的询问选是，不提示

-f 尝试修正系统依赖损坏处

-m 如果归档无法定位，尝试继续

-u 同时显示更新软件包的列表

-b 获取源码包后编译

-V 显示详细的版本号

-c=? 阅读此配置文件

-o=? 设置自定的配置选项，如 -o dir::cache=/tmp

常用命令：

1）apt-get update 更新源

【aptitude update】

2）apt-get dist-upgrade 升级系统到相应的发行版(根据 source.list 的配置)

【aptitude dist-upgrade】

3）apt-get upgrade 更新所有已经安装的软件包

【aptitude upgrade】

4）apt-get install package\_name 安装软件包(加上 --reinstall重新安装)

apt-get install package\_name=version 安装指定版本的软件包

【aptitude install package\_name】

5）apt-get remove package\_name 卸载一个已安装的软件包（保留配置文件）

【aptitude remove package\_name】

6）apt-get purge package\_name 移除软件包（删除配置信息）

或apt-get --purge remove packagename

【aptitude purge package\_name】

7）apt-get check 检查是否有损坏的依赖

8）apt-get autoclean   
删除你已经删掉的软件（定期运行这个命令来清除那些已经卸载的软件包的.deb文件。通过这种方式，您可以释放大量的磁盘空间。如果您的需求十分迫切，可以使用apt-get  
clean以释放更多空间。这个命令会将已安装软件包裹的.deb文件一并删除。大多数情况下您不会再用到这些.debs文件，因此如果您为磁盘空间不足  
而感到焦头烂额，这个办法也许值得一试）

【aptitude autoclean】

9）apt-get clean 把安装的软件的备份也删除，不过这样不会影响软件的使用

【aptitude clean】

2、apt-cache

apt-cache - query the APT cach.

apt-cache performs a variety of operations on APT's package cache. apt-cache does not manipulate the state of the system but does provide operations to search and generate interesting output from the package metadata.

1）apt-cache depends packagename 了解使用依赖

2）apt-cache rdepends packagename 是查看该包被哪些包依赖

3）apt-cache search packagename 搜索包

【aptitude search packagename】

4）apt-cache show packagename 获取包的相关信息，如说明、大小、版本等

【aptitude show packagename】

5）apt-cache showpkg packagename 显示软件包的大致信息

（注：中括号【】内的aptitude也是类似于apt-\*的一个包管理上层工具）