# Ubuntu/Debian Linux apt-get package management cheat sheet

Both Debian and Ubuntu Linux provide a number of package management tools.
This article summaries package management command along with its usage and examples for you.

**apt-get** : APT is acronym for Advanced Package Tool. It supports installing packages over internet using ftp or http protocols. You can also upgrade all packages in a single operation, which makes it even more attractive. For scripting purpose apt-get is perfect tool.

- **dpkg** : Debian packaging tool which can be used to install, query, uninstall packages.
- **apt**:
- Gui tools: You can also try GUI based or high-level interface to the Debian GNU/Linux package system. Following list summaries, them:
  - aptitude: It is a text-based interface to the Debian GNU/Linux package system.
  - synaptic: GUI front end for APT

## Understanding .deb file

Red hat Linux package names generally end in .rpm, similarly Debian package names end in .deb, for example:

```
apache_1.3.31-6_i386.deb
```

Where,

1. `apache` : Package name
2. `1.3.31-6` : Version number

3. `i386` : Hardware Platform on which this package will run (i386 == intel x86 based system)
4. `.deb` : Extension that suggest it is a Debian package

Remember, whenever I refer **.deb** file it signifies complete file name, and whenever I refer package name it must be first part of .deb file. For example, when I refer to a package sudo it means sudo only and not the .deb file i.e. sudo_1.6.7p5-2_i386.deb. You can find out debian package name with the following command:

```
apt-cache search {package-name}
```

```
apt-cache search apache
```

Finally, most of the actions listed in this post are written with the assumption that they will be executed by the root user running the bash or any other modern shell. Otherwise add sudo command before apt-get:

```
$ sudo apt-get ....
```

Ok, let us see some examples.

# apt-get add a new package

The syntax is:

```
apt-get install {package-name}
```

To install a package called samba, run:

```
# apt-get install samba
```

OR

```
$ sudo apt-get install samba
```

## How do I search for package names?

To find software packages use the apt-cache command:

```
$ apt-cache search {package-name}
```

To find vim package list, enter:

```
$ apt-cache search vim
```

OR

```
$ apt-cache search vim | more
```

OR

```
$ apt-cache search vim | grep 'word'
```

You can limit search with regex too. For example, show all vim packages starting with vim word only:

```
$ apt-cache search ^vim
```

Sample outputs:

```
vim - Vi IMproved - enhanced vi editor

vim-common - Vi IMproved - Common files

vim-doc - Vi IMproved - HTML documentation

vim-gnome - Vi IMproved - enhanced vi editor (dummy package)

vim-gtk3 - Vi IMproved - enhanced vi editor - with GTK3 GUI

vim-gui-common - Vi IMproved - Common GUI files

....

..

...

vim-vimerl - Erlang plugin for Vim

vim-vimerl-syntax - Erlang syntax for Vim
```

```
vim-vimoutliner - script for building an outline editor on top of Vim

vim-voom - Vim two-pane outliner

vim-youcompleteme - fast, as-you-type, fuzzy-search code completion
engine for Vim
```

## apt-get remove the package called samba but keep the configuration files

The syntax is:

```
apt-get remove {package-name}
```

For example, to remove a package named samba, run:

```
# apt-get remove samba
```

## apt-get remove (erase) package and configuration files too

The syntax is:

```
apt-get --purge remove {package-name}
```

For example, delete a package named samba including config files stored in /etc/ directory:

```
# apt-get --purge remove samba
```

## How do I remove packages that were automatically installed to satisfy

## dependencies for other packages and are now no longer needed?

Run the following command:

```
$ sudo apt-get autoremove
```

OR

```
$ sudo apt-get --purge autoremove
```

The above command is quite useful to remove unwanted kernel on a Debian or Ubuntu Linux.

## How do I update my system?

The syntax is:

```
# apt-get update
# apt-get upgrade
```

To upgrade individual package called sudo or bash, enter:

```
# apt-get install sudo

# apt-get install bash
```

Sample outputs:

```
Reading package lists... Done

Building dependency tree

Reading state information... Done

Suggested packages:

  bash-doc
```

```
Recommended packages:

  bash-completion

The following packages will be upgraded:
  bash
1 upgraded, 0 newly installed, 0 to remove and 7 not upgraded.
Need to get 1427 kB of archives.
After this operation, 0 B of additional disk space will be used.
Get:1 http://cdn-fastly.deb.debian.org/debian stretch/main amd64 bash
amd64 4.4-4+b2 [1427 kB]
Fetched 1427 kB in 2s (661 kB/s)
debconf: delaying package configuration, since apt-utils is not installed
(Reading database ... 8417 files and directories currently installed.)
Preparing to unpack .../bash_4.4-4+b2_amd64.deb ...
Unpacking bash (4.4-4+b2) over (4.4-4+b1) ...
Setting up bash (4.4-4+b2) ...
```

## apt-get display available software updates

Type the following command to display the list of all available upgrades (updates) using -u option

```
# apt-get update
# apt-get -u upgrade
```

Sample outputs:

```
Reading package lists... Done

Building dependency tree

Reading state information... Done

Calculating upgrade... Done

The following packages will be upgraded:

  apt bash gcc-6-base init init-system-helpers libapt-pkg5.0 libgcc1 libstdc++6

8 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.

Need to get 4244 kB of archives.

After this operation, 0 B of additional disk space will be used.

Do you want to continue? [Y/n]
```

If you decided to upgrade all of the shown packages just hit ‘y’ key. If you just want to see a list, enter:

```
# apt-get -u upgrade --assume-no
```

However, if you just wish to upgrade individual package then use apt-get command and it will take care of rest of your worries:

```
# apt-get install {package-name}
```

## How do I upgrade my Debian or Ubuntu Linux distro?

From the man page:

The dist-upgrade option to apt-get in addition to performing the function of upgrade, also intelligently handles changing dependencies with new versions of packages; apt-get has a “smart” conflict resolution system, and it will attempt to upgrade the most important packages at the expense of less important ones if necessary. The dist-upgrade command may therefore remove some packages.

This is useful to upgrade your distro including kernel:

```
$ sudo apt-get update

$ sudo apt-get dist-upgrade
```

## dpkg command to get package information such as description of package, version etc.

The syntax is:

```
dpkg --info {.deb-package-name}
```

For example:

```
$ dpkg --info sudo_1.6.7p5-2_i386.deb | less
```

## List all installed packages

The syntax is:

```
dpkg -l
```

Example:

```
$ dpkg -l
```

## How do I list individual package such as apache or sudo

```
$ dpkg -l apache
```

Sample outputs:

```
Desired=Unknown/Install/Remove/Purge/Hold

| Status=Not/Inst/Conf-files/Unpacked/halF-conf/Half-inst/trig-aWait/Trig-
pend

|/ Err?=(none)/Reinst-required (Status,Err: uppercase=bad)

||/ Name                    Version         Architecture        Description
```

```
+++-=====================================-
=======================-=======================-
===============================================================
=================


ii  sudo                          1.8.19p1-1ubuntu1      amd64               Provide
limited super user privileges to specific users
```

You can also use this command to see (verify) if package sudo is installed or not (note that if package is installed then it displays package name along with small description):

```
$ dpkg -l | grep -i 'sudo'
```

To list packages related to the apache:

```
$ dpkg -l '*apache*'
```

List files provided (or owned) by the installed package (for example what files are provided by the installed samba package). The syntax is:

```
dpkg -L {package-name}
```

For example:

```
$ dpkg -L samba

$ dpkg -L sudo
```

Sample outputs:

```
/.


/etc


/etc/pam.d


/etc/pam.d/sudo


/etc/sudoers


/etc/sudoers.d
```

```
/etc/sudoers.d/README
/lib
/lib/systemd
/lib/systemd/system
/usr
/usr/bin
....
..
...
/var/run
/usr/bin/sudoedit
/usr/lib/sudo/libsudo_util.so
/usr/lib/sudo/libsudo_util.so.0
/usr/share/man/man8/sudoedit.8.gz
```

To list files provided (or owned) by the package (for example what files are provided by the uninstalled sudo package). The syntax is:

```
dpkg --contents {.deb-package-name}
```

For example:

```
# dpkg --contents sudo_1.6.7p5-2_i386.deb
```

## Find out what package owns the file /bin/netstat?

The syntax is:

```
dpkg -S {/path/to/file}
```

For example:

```
$ dpkg -S /bin/netstat
```

Sample outputs:

```
net-tools: /bin/netstat
```

So net-tools package provided /bin/netstat command.

## How do I search for package or package description?

Sometimes you don’t know package name but aware of some keywords to search the package. Once you got package name you can install it using apt-get -i {package-name} command:

```
apt-cache search "Text-to-search"

apt-cache search "httpd"

apt-cache search "web server"

apt-cache search "web server" | grep something
```

### Find out all the Debian package which can be used for intrusion detection

Type the following command:

```
code>$ apt-cache search "Intrusion Detection"
```

Find out all sniffer packages, run:

```
$ apt-cache search sniffer
```

## Find out if Debian package is installed or not (status)

The syntax is:

```
dpkg -s {package-name} | grep Status
```

For example:

```
$ dpkg -s samba| grep Status
```

## How do I list each dependency a package has...

Display a listing of each dependency a package has and all the possible other packages that can fulfill that dependency. You hardly use this command as apt-get does decent job fulfill all package dependencies. The syntax is:

```
apt-cache depends package
```

To display dependencies for lsof and mysql-server packages:

```
$ apt-cache depends mysql-server
```

```
$ apt-cache depends lsof
```

Sample outputs:

```
Lsof


 Depends: libc6
```

```
Depends: libselinux

Suggests: perl
```

***************************************************************************

## dpkg command cheat sheet for Debian and Ubuntu Linux

dpkg is a package manager for Debian, Ubuntu and many other Linux distro. It is used to install/manage individual packages. Here is a quick cheat sheet you will find handy while using dpkg at shell prompt:

| Syntax | Description | Example |
|---|---|---|
| dpkg -i {.deb package} | Install the package | dpkg -i zip_2.31-3_i386.deb |
| dpkg -i {.deb package} | Upgrade package if it is installed else install a fresh copy of package | dpkg -i zip_2.31-3_i386.deb |
| dpkg -R {Directory-name} | Install all packages recursively from directory | dpkg -R /tmp/downloads |
| dpkg -r {package} | Remove/Delete an installed package except configuration files | dpkg -r zip |
| dpkg -P {package} | Remove/Delete everything including configuration files | dpkg -P apache-perl |
| dpkg -l | List all installed packages, along with package version and short description | dpkg -l<br>dokg -l \| less<br>dpkg -l '*apache*'<br>dpkg -l \| grep -i 'sudo' |
| dpkg -l {package} | List individual installed packages, along with package version and short description | dpkg -l apache-perl |
| dpkg -L {package} | Find out files are provided by the installed package i.e. list where files were installed | dpkg -L apache-perl<br>dpkg -L perl |
| dpkg -c {.Deb package} | List files provided (or owned) by the package i.e. List all files inside debian .deb package file, very useful to find where files would be installed | dpkg -c dc_1.06-19_i386.deb |
| dpkg -S {/path/to/file} | Find what package owns the file i.e. find out what package does file belong | dpkg -S /bin/netstat<br>dpkg -S /sbin/ippool |
| dpkg -p {package} | Display details about package group, version, maintainer, Architecture, display depends packages, description etc | dpkg -p lsof |
| dpkg -s {package} \| grep Status | Find out if Debian package is installed or not (status) | dpkg -s lsof \| grep Status |

{package} - Replace with actual package name

*********************************************************************************************

### apt-get command examples cheat sheet for Debian Linux

**********************************************************************************

## apt-get is the command-line tool for handling packages for Debian Linux which is use to:

- Install/manage individual packages
- Upgrade packages
- Apply security patch(s)
- Keep Debian system up to date
- Download source .deb files
- Has many GUI and other utilities as front-ends

Here is quick cheat sheet you will find handy while using apt-get at shell prompt:

| Syntax | Description | Example(s) |
|---|---|---|
| apt-get install {package} | Install the new package. If package is installed then try to upgrade to latest version | apt-get install zip<br>apt-get install lsof samba mysql-client |
| apt-get remove {package} | Remove/Delete an installed package except configuration files | apt-get remove zip |
| apt-get --purge remove {package} | Remove/Delete everything including configuration files | apt-get --purge remove mysql-server |
| apt-get update<br>apt-get upgrade | Resynchronize the package index files and Upgrade the Debian Linux system including security update (Internet access required) | apt-get update<br>apt-get upgrade |
| apt-get update<br>apt-get dist-upgrade | Usually use to upgrade to Debian distribution. For example Woody to Sarge upgrade. 'dist-upgrade' in addition to performing the function of upgrade, also intelligently handles changing dependencies with new versions of packages; apt-get has a "smart" conflict resolution system, and it will attempt to upgrade the most important packages at the expense of less important ones if necessary. | apt-get update<br>apt-get dist-upgrade |

For more info see:

Disclaimer: Above details extracted from Open-Source