APT Tips

Ubunto or Debian or other Linux distributions are using APT (Advanced Packaging Tool) as default software management utility.

About APT

The original APT was implemented for Debian GNU/Linux. The RPM port of APT was made by Connectiva Linux. In addition to the above few (either home-grown or written elsewhere) extensions and wrappers were added. In the default setup the system automatic software updates mechanism is enabled.

APT links

* APT Home page (Obsolete documentation)
* APT How To

Enabling/Disabling update system

In order to enable the automatic update system run following commands:

```
chkconfig --add apt-autoupdate
service apt-autoupdate start
```

In order to disable the system run (**not reccomended!**):

```
service apt-autoupdate stop
chkconfig --del apt-autoupdate
```

Updating packages

To update installed software set to latest versions run:

```
apt-get update
```

which will synchronize local repository information with the remote one.

```
apt-get upgrade
```

which will perform necessary dependency resolution, download packages and install them.

Smart upgrade (dist-upgrade)

In some case the standard upgrade method is not enough to upgrade your system (for example when some software is repackaged, dependencies in new version include new software, etc.). In such case the 'smart' upgrade should be used. Run:

```
apt-get update
```

which will synchronize local repository information with the remote one.

```
apt-get dist-upgrade
```

which will perform the 'smart' dependency resolution, download and install necessary packages (but also may remove obsolete packages).

Updating kernel packages

Kernel packages (and depending on them kernel-module packages) are a special case: These are never upgraded but rather installed in parallel to the existing ones. In order to upgrade the kernel run:

```
apt-get update
```

which will synchronize local repository information with the remote one.

```
apt-get install {kernel | kernel-smp}
```

which will download and install kernel packages. **A reboot is required to use the new kernel.**

Excluding packages from upgrade

The upgrade,dist-upgrade and upgrade-kernel operations will apply all upgrades available from the package repositories. If for some reason some packages should be excluded from such upgrade a per-package apt configuration file can be created this way:

```
echo "RPM::Ignore:: \"PACKAGENAME\";" > /etc/apt/apt.conf.d/PACKAGENAME.conf
```

(Please note that above configuration setting can be also done in main apt configuration file, but keeping it as a separate configuration file eases the maintenance). Please note that packages depending on 'ignored' package will also be excluded from future upgrades. To restore default system upgrade procedure please remove ///etc/apt/apt.conf.d/rpm-PACKAGENAME.conf// file.

How to keep specific versions of packages installed

You may have occasion to modify something in a package and don't have time or don't want to port those changes to a new version of the program. You can "pin" the version you have installed so that it will not be upgraded. Using this resource is simple. You just need to edit the file ///etc/apt/preferences//.

The format is simple:

```
    Package: <package>
    Pin: <pin definition>
    Pin-Priority: <pin's priority>
```

The **pin priority** helps determine whether a package matching the "Packages:" and "Pin:" lines will be installed, with higher priorities making it more likely that a matching package will be installed. You can read //apt_preferences// for a thorough discussion of priorities, but a few examples should give the basic idea. The following describes the effect of setting the priority field to different values in the sylpheed example above.

* **1001** A package will never be replaced by apt. If available, apt will install version x.y.z even if it would replace an installed package with a higher version. Only packages of priority greater than 1000 will ever downgrade an existing package.

* **1000** The effect is the same as priority 1001, except that apt will refuse to downgrade an installed version to x.y.z

* **990** Version x.y.z will be replaced only by a higher version available from a release designated as preferred using the "APT::Default-Release" variable.

* **500** Any version higher than x.y.z of a package which is available from any release will take preference over version x.y.z, but x.y.z will still be preferred to a lower version.

* **100** Higher versions of a package available from any release will take preference over version x.y.z, as will any installed higher version of the package; so x.y.z will be installed only if no version is installed already. This is the priority of installed packages.

* **-1** Negative priorities are allowed as well, and preventx.y.z from ever being installed.

pin can be specified on a package's //version//, //release// or //origin//.

Checking system consistency

In order to check system consistency run:

```
apt-get check
```

This topic: IGIRelease > APTTips
Topic revision: r2 - 2012-01-04 - CristinaAiftimiei