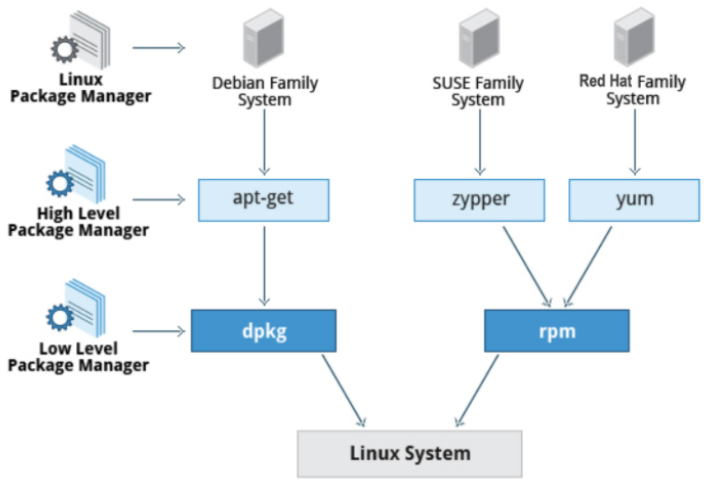
Installing Software

Package Management Systems on Linux:

1. The core parts of a Linux distribution and most of its add-on software are installed via the Package Management System.
2. Each package contains the files and other instructions needed to make one software component work well and cooperate with the other components that comprise the entire system.
3. Packages can depend on each other.
4. For example, a package for a web-based application written in **PHP** can depend on the **PHP** package.
5. There are two broad families of package managers: those based on **Debian** and those which use **RPM** as their low-level package manager.
6. The two systems are incompatible, but broadly speaking, provide the same features and satisfy the same needs.

Package Managers - Two Levels:

1. Both package management systems operate on two distinct levels
2. A low-level tool (such as **dpkg** or **rpm**) takes care of the details of unpacking individual packages, running scripts, getting the software installed correctly
3. A high-level tool (such as **apt-get**, **dnf**, **yum**, or **zypper**) works with groups of packages, downloads packages from the vendor, and figures out dependencies.
4. Most of the time users need to work only with the high-level tool, which will take care of calling the low-level tool as needed.
5. Dependency resolution is a particularly important feature of the high-level tool, as it handles the details of finding and installing each dependency for you.
6. However, as installing a single package could result in many dozens or even hundreds of dependent packages being installed.



Package Managers: Two Levels

Working With Different Package Management Systems:

1. The Advanced Packaging Tool (**apt**) is the underlying package management system that manages software on Debian-based systems.
2. While it forms the backend for graphical package managers, such as the Ubuntu Software Center and synaptic, its native user interface is at the command line, with programs that include apt (or **apt-get**) and **apt-cache**.
3. **dnf** is the open source command-line package-management utility for the **RPM**-compatible Linux systems that belongs to the Red Hat family.
4. **dnf** has both command line and graphical user interfaces.
5. **Fedora** and **RHEL 8** replaced the older yum utility with **dnf**
6. **zypper** is the package management system for the **SUSE/openSUSE** family and is also based on **RPM**.
7. **zypper** also allows you to manage repositories from the command line.
8. **zypper** is fairly straightforward to use and resembles **dnf/yum** quite closely.

Operation	rpm	deb
Install package	`rpm -i foo.rpm`	`dpkg --install foo.deb`
Install package, dependencies	`dnf install foo`	`apt-get install foo`
Remove package	`rpm -e foo.rpm`	`dpkg --remove foo.deb`
Remove package, dependencies	`dnf remove foo`	`apt-get autoremove foo`
Update package	`rpm -U foo.rpm`	`dpkg --install foo.deb`
Update package, dependencies	`dnf update foo`	`apt-get install foo`
Update entire system	`dnf update`	`apt-get dist-upgrade`
Show all installed packages	`rpm -qa` or `dnf list installed`	`dpkg --list`
Get information on package	`rpm -qil foo`	`dpkg --listfiles foo`
Show packages named `foo`	`dnf list "foo"`	`apt-cache search foo`
Show all available packages	`dnf list`	`apt-cache dumpavail foo`
What package is `file` part of?	`rpm -qf file`	`dpkg --search file`