Package Management Basics

apt, yum, dnf, zypper, and pkg

Table of Contents

Abstract

This guide is intended as a quick reference for the fundamentals of finding, installing, and upgrading packages on a variety of distributions, and should help you translate that knowledge between systems.

Audience

For those new to Linux who need a basic understanding of package management.

Original version of this doc

The original version of this guide can be found at Digital Ocean [https://www.digitalocean.com/community/tutorials/package-management-basics-apt-yum-dnf-pkg].

Revision History

15.1.2016	v1.0 Converted and edited for TLDP	Jason Evans
15.2.2016	v1.1 Changed format to asciidoc and made corrections	Jason Evans
30.3.2016	v1.2 Added documentation for SuSE, Abstract section,	Jason Evans

	and expanded overview, also many small typographical changes and corrections.	

Contributions

- Brennen Bearnes [https://www.digitalocean.com/community/users/bpb] (original author).
- Jason Evans [http://wiki.tldp.org/Jason%20Evans] (editor and maintainer for TLDP)

Feedback

Missing information, missing links, missing characters? Mail it to the maintainer of this document: jsevans *at* youvegotthe.info

Copyright information

Introduction

Why was this document written?

Most modern Unix-like operating systems offer a centralized mechanism for finding and installing software. Software is usually distributed in the form of packages, kept in repositories. Working with packages is known as package management. Packages provide the basic components of an operating system, along with shared libraries, applications, services, and documentation.

A package management system does much more than one-time installation of software. It also provides tools for upgrading already-installed packages. Package repositories help to ensure that code has been vetted for use on your system, and that the installed versions of software have been approved by developers and package maintainers.

When configuring servers or development environments, it's often necessary look beyond official repositories. Packages in the stable release of a distribution may be out of date, especially where new or rapidly-changing software is concerned. Nevertheless, package management is a vital skill for system administrators and developers, and the wealth of packaged software for major distributions is a tremendous resource.

What do you need?

This guide covers Debian, Ubuntu, CentOS, Fedora, SuSE, and FreeBSD and will require one of those distributions to be installed.

Note

All of the commands in this guide assume that the user is running the commands as root or with `sudo`.

Package Management Systems: A Brief Overview

In a Windows environment, programs are packaged in .exe or .msi installers which will then install most of the files needed to run the program. If your computer doesn't have some dependant applications, then the program that you are trying to run will either not install or not run properly. You will then have to scour the internet in order to find the missing required applications or libraries. For example in CentOS 7, in order to install the VIM text editor, I need to add the following packages:

```
gpm-libs                    x86_64     1.20.7-5.el7          base      32 k
groff-base                  x86_64     1.22.2-8.el7          base     942 k
perl                        x86_64     4:5.16.3-286.el7      base     8.0 M
perl-Carp                   noarch     1.26-244.el7          base      19 k
perl-Encode                 x86_64     2.51-7.el7            base     1.5 M
perl-Exporter               noarch     5.68-3.el7            base      28 k
perl-File-Path              noarch     2.09-2.el7            base      26 k
perl-File-Temp              noarch     0.23.01-3.el7         base      56 k
perl-Filter                 x86_64     1.49-3.el7            base      76 k
perl-Getopt-Long            noarch     2.40-2.el7            base      56 k
perl-HTTP-Tiny              noarch     0.033-3.el7           base      38 k
perl-PathTools              x86_64     3.40-5.el7            base      82 k
perl-Pod-Escapes            noarch     1:1.04-286.el7        base      50 k
perl-Pod-Perldoc            noarch     3.20-4.el7            base      87 k
perl-Pod-Simple             noarch     1:3.28-4.el7          base     216 k
perl-Pod-Usage              noarch     1.63-3.el7            base      27 k
perl-Scalar-List-Utils      x86_64     1.27-248.el7          base      36 k
perl-Socket                 x86_64     2.010-3.el7           base      49 k
perl-Storable               x86_64     2.45-3.el7            base      77 k
perl-Text-ParseWords        noarch     3.29-4.el7            base      14 k
perl-Time-HiRes             x86_64     4:1.9725-3.el7        base      45 k
perl-Time-Local             noarch     1.2300-2.el7          base      24 k
perl-constant               noarch     1.27-2.el7            base      19 k
perl-libs                   x86_64     4:5.16.3-286.el7      base     687 k
perl-macros                 x86_64     4:5.16.3-286.el7      base      43 k
perl-parent                 noarch     1:0.225-244.el7       base      12 k
perl-podlators              noarch     2.5.1-3.el7           base     112 k
perl-threads                x86_64     1.87-4.el7            base      49 k
perl-threads-shared         x86_64     1.43-6.el7            base      39 k
vim-common                  x86_64     2:7.4.160-1.el7       base     5.9 M
vim-filesystem              x86_64     2:7.4.160-1.el7       base     9.6 k
which                       x86_64     2.20-7.el7            base      41 k
```

Imagine trying to manually install all of these programs one at a time just to be able to install a text editor! In the early days of Linux, we faced these kinds of problems, however this problem is fixed with package management systems such as apt, yum, and others. Package managers simplify everything. They look at the package that you want to install such as VIM, LibreOffice, etc., then look at what other package it depends upon, the dependencies of those packages, and so on; then it downloads them all and installs them. For example, in order to install VIM in CentOS 7 today, I simply have to run `yum install vim`.

Some different package management systems:

While their functionality and benefits are broadly similar, packaging formats and tools vary by platform:

Operating System	Format	Tool(s)
Debian	.deb	apt, apt-cache, apt-get, dpkg

Operating System	Format	Tool(s)
Ubuntu	.deb	apt, apt-cache, apt-get, dpkg
CentOS	.rpm	yum
Fedora	.rpm	dnf
SuSE	.rpm	zypper
FreeBSD	Ports, .txz	make, pkg

In Debian and systems based on it such as Ubuntu, Linux Mint, and Raspbian, the package format is the .deb file. Apt, the Advanced Packaging Tool, provides commands used for most common operations: Searching repositories, installing collections of packages and their dependencies, and managing upgrades. APT commands operate as a front-end to the lower-level dpkg utility, which handles the installation of individual .deb files on the local system, and is sometimes invoked directly.

Fedora and enterprise level distributions like Red Hat Enterprise Linux (RHEL), CentOS, and Oracle Linux use RPM files. In CentOS, Oracle, and RHEL, yum is used to interact with both individual package files and repositories. In recent versions of Fedora, yum has been replaced by dnf, a modernized fork which retains most of yum's functionality.

SuSE also uses RPM files. However, the package management software is known as zypper. Zypper's command line interface is very similar to yum. SuSE also has a build-in gui called YasT that can handle package management and can be accessed from a graphical mode or from the command line.

FreeBSD's binary package system is administered with the pkg command. FreeBSD also offers the Ports Collection, a local directory structure and tools which allow the user to fetch, compile, and install packages directly from source using Makefiles. It's usually much more convenient to use pkg, but occasionally a pre-compiled package is unavailable, or syou may need to change compile-time options.

Update Package Lists

Most systems keep a local database of the packages available from remote repositories. It's best to update this database before installing or upgrading packages. As a partial exception to this pattern, yum and dnf will check for updates before performing some operations, but you can ask them at any time whether updates are available.

System	Command
Debian / Ubuntu	`apt-get update`
CentOS	`yum check-update`
Fedora	`dnf check-update`
SuSE	`zypper refresh`
FreeBSD Packages	`pkg update`
FreeBSD Ports	`portsnap fetch update`

Upgrade Installed Packages

Making sure that all of the installed software on a machine stays up to date would be an enormous undertaking without a package system. You would have to track upstream changes and security alerts for hundreds of different packages. While a package manager doesn't solve every problem you'll encounter when upgrading software, it does enable you to maintain most system components with a few commands.

On FreeBSD, upgrading installed ports can introduce breaking changes or require manual configuration steps. It's best to read /usr/ports/UPDATING before upgrading with portmaster.

System	Command	Notes
Debian / Ubuntu	`apt-get upgrade`	Only upgrades installed packages, where possible.
	`apt-get dist-upgrade`	May add or remove packages to satisfy new dependencies.
CentOS	`yum update`	
Fedora	`dnf upgrade`	
SuSE	`zypper update`	
FreeBSD Packages	`pkg upgrade`	
FreeBSD Ports	`less /usr/ports/UP-DATING`	Uses less to view update notes for ports (use arrow keys to scroll, pressq to quit).
	`cd /usr/ports/ports-mgmt/portmaster && make install && port-master -a`	Installs portmaster and uses it to update installed ports.

Find a Package

Most distributions offer a graphical or menu-driven front end to package collections. These can be a good way to browse by category and discover new software. Often, however, the quickest and most effective way to locate a package is to search with command-line tools.

System	Command	Notes
Debian / Ubuntu	`apt-cache search`	
CentOS	`yum search`	
	`yum search all`	Searches all fields, in-cluding description.
Fedora	`dnf search`	
	`dnf search all`	Searches all fields, in-cluding description.
SuSE	`zypper se`	
FreeBSD Packages	`pkg search`	Searches by name.
	`pkg search -f`	Searches by name, re-turning full descriptions.
	`pkg search -D`	Searches description.
FreeBSD Ports	`cd /usr/ports && make search name=package`	Searches by name.
	`cd /usr/ports && make search key=`	Searches comments, descrip-tions, and dependencies.

View Info About a Specific Package

When deciding what to install, it’s often helpful to read detailed descriptions of packages. Along with human-readable text, these often include metadata like version numbers and a list of the package’s depen-dencies.

System	Command	Notes
Debian / Ubuntu	`apt-cache show package`	Shows locally-cached info about a package.
	`dpkg -s package`	Shows the current installed status of a package.
CentOS	`yum info package`	
	`yum deplist package`	Lists dependencies for a package.
Fedora	`dnf info package`	
	`dnf repoquery -\/-requires package`	Lists dependencies for a package.
SuSE	`zypper info search string`	Lists dependencies for a package.
FreeBSD Packages	`pkg info package`	Shows info for an installed package.
FreeBSD Ports	`cd /usr/ports/category/port && cat pkg-descr`	

Install a Package from Repositories

Once you know the name of a package, you can usually install it and its dependencies with a single command. In general, you can supply multiple packages to install simply by listing them all.

System	Command	Notes
Debian / Ubuntu	`apt-get install package`	
	`apt-get install package1 package2`	Installs all listed packages.
	`apt-get install -y package`	Assumes "yes" where apt would usually prompt to continue.
CentOS	`yum install package`	
	`yum install package1 package2`	Installs all listed packages.
	`yum install -y package`	Assumes "yes" where yum would usually prompt to continue.
Fedora	`dnf install package`	
	`dnf install package1 package2`	Installs all listed packages.
	`dnf install -y package`	Assumes "yes" where dnf would usually prompt to continue.
SuSE	`zypper install`	
FreeBSD Packages	`pkg install package`	
	`pkg install package1 package2`	Installs all listed packages.

System	Command	Notes
FreeBSD Ports	`cd /usr/ports/category/port && make install`	Builds and installs a port from source.

Install a Package from the Local Filesystem

Sometimes, even though software isn't officially packaged for a given operating system, a developer or vendor will offer package files for download. You can usually retrieve these with your web browser, or viacurl on the command line. Once a package is on the target system, it can often be installed with a single command.

On Debian-derived systems, dpkg handles individual package files. If a package has unmet dependencies, gdebi can often be used to retrieve them from official repositories.

On CentOS and Fedora systems, yum and dnf are used to install individual files, and will also handle needed dependencies.

System	Command	Notes
Debian / Ubuntu	`dpkg -i package.deb`	
	`apt-get install -yg debi && gdebi package.deb`	Installs and uses gdebi to install package.deb and retrieve any missing dependencies.
CentOS	`yum install package.rpm`	
Fedora	`dnf install package.rpm`	
SuSE	`zypper install package.rpm`	
FreeBSD Packages	`pkg add package.txz`	
	`pkg add -f package.txz`	Installs package even if already installed.

Remove One or More Installed Packages

Since a package manager knows what files are provided by a given package, it can usually remove them cleanly from a system if the software is no longer needed.

System	Command	Notes
Debian / Ubuntu	`apt-get remove package`	yum remove package
	`apt-get autoremove`	Removes unneeded packages.
CentOS	`yum remove package`	
Fedora	`dnf erase package`	
FreeBSD Packages	`pkg delete package`	
	`pkg autoremove`	Removes unneeded packages.
SuSE	`zypper rm package`	Removes unneeded packages.
FreeBSD Ports	`pkg delete package`	

System	Command	Notes
	`cd /usr/ports/path_to_port && make deinstall`	De-installs an installed port.

Get Help

In addition to web-based documentation, keep in mind that Unix manual pages (usually referred to as man pages) are available for most commands from the shell. To read a page, use the `man` command. For example, `man yum` will give you a brief manual on how to use yum.

Conclusion and Further Reading

This guide provides an overview of basic operations that can be cross-referenced between systems, but only scratches the surface of a complex topic. For greater detail on a given system, you can consult the following resources:

1. This guide [https://www.digitalocean.com/community/tutorials/ubuntu-and-debian-package-management-essentials] covers Ubuntu and Debian package management in detail.
2. There's an official CentOS guide to managing software with yum [https://www.centos.org/docs/5/html/yum/].
3. There's a Fedora wiki page about dnf [https://fedoraproject.org/wiki/Dnf], and an official manual for dnf itself [https://dnf.readthedocs.org/en/latest/index.html]
4. This guide [https://www.digitalocean.com/community/tutorials/how-to-manage-packages-on-freebsd-10-1-with-pkg] covers FreeBSD package management using pkg.
5. The FreeBSD Handbook [https://www.freebsd.org/doc/handbook/] contains a section on using the Ports Collection [https://www.freebsd.org/doc/handbook/ports-using.html].
6. OpenSuSE documentation for Zypper [http://doc.opensuse.org/documentation/html/openSUSE_114/opensuse-reference/cha.sw_cl.html] and YaST [http://doc.opensuse.org/documentation/html/openSUSE_114/opensuse-reference/cha.onlineupdate.you.html].