6.3.3 Managing Debian Packages

Click one of the buttons to take you to that part of the video.

Manage Debian Packages 0:00-0:20

In this demonstration we're going to talk about managing Debian packages. There are two commands that you can use to manage Debian packages. We can use the dpkg command or we can use the apt commands. Let's begin by looking at using the dpkg command to manage Debian packages.

View Packages Using dpkg 0:21-1:27

For example, if we wanted to see if a particular package is installed on the system already and what its status is, we can use the 'dpkg' command with the '-p' option, followed by the name of the package in question.

For example, we can use this command to see whether or not the 'gcc' package is installed on this system. gcc is a C compiler. It's very commonly installed on Linux distributions and it is used to compile source code into an executable that can be run on the system.

Before I run this command, I do need to point out that you must use a lowercase p right here. The reason you must use a lowercase p and not an uppercase P is because uppercase P uninstalls whatever package you specify right here, not display your information about it, and that's not what we want to do.

I'll go ahead and press Enter with a lowercase p, and we will view information about the package. We see that it's installed on the system, we see the package name, we see what version number is currently installed of the package, we see what functionality it provides, we see what other packages this package is dependent upon as well, and we also see the size of the package.

Install Packages Using dpkg 1:28-2:57

You can also use the dpkg command to install a new Debian package on the system. I've already pulled down a couple of Debian packages. They're in my Downloads directory of my user account. I'll do an 'ls' command.

Let's install this package right here, eboard. eboard is a graphical chess game that you can run on Linux to play chess. We can use the dpkg command to install this package, but we cannot do it as my rtracy user account. Rtracy is allowed to view package information, but rtracy is not allowed to install new packages. We have to use the sudo command in order to gain root level privileges to install packages.

We'll do 'sudo dpkg -i' for install, and then we want to install the eboard package name. I just used the handy tab-complete feature of the bash shell, so I don't have to type out the full name manually. Hit Enter and the package is installed.

If I wanted to, I could run the same command that we looked at before, 'dpkg -p', but this time look at 'eboard'. We learn that it's installed. We view its version number and we can view description of it.

This is a chess board program. In fact, we can actually run it. We can type 'eboard' from the command prompt and there is my chess game that's running. I'll go ahead and close out of that.

Uninstall Packages Using dpkg 2:58-3:47

Just as you can install a new package with the dpkg command, you can also uninstall a package with the dpkg command. Let's, once again, use 'sudo' to get root level permissions to the system. We'll run 'dpkg'. This time we'll use '-P' which uninstalls the package that you specify. We'll just simply type its name, 'eboard', Enter, and the eboard package is removed.

dpkg works fine for installing and uninstalling packages, but it's got one glaring weakness. That is the fact that dpkg cannot resolve dependencies. If you try to install a package that has a long chain of many dependencies in order to run, you'll spend a lot of time chasing down all of those dependencies of trying to manually install them.

Use apt Commands 3:48-4:03

A better option for installing Debian packages on a system is to use the apt commands instead. In fact, using apt is the preferred way to manage Debian packages. It makes life so much easier. In fact, I don't ever use the dpkg command. I always use apt commands instead.

Use apt-cache 4:04-6:38

For example, you can use the 'apt-cache' command to manage packages that are already installed on your system. For example, if we just want to see whether or not a package has been installed, we can run 'apt-cache showpkg', and then the name of the package in question.

Let's look at 'gcc' again, our C compiler.

In the output of the apt-cache command, we can see several pieces of key information about the gcc package. We see its version number right here. We see the Reverse Depends. These are the other packages on the system that require gcc to be installed in order for them to run.

Then down here under Dependencies, we see the packages that gcc is dependent upon in order for it to run. Down here under provides, we can see what functionality gcc provides. It's a C compiler.

You can also run the 'apt-cache' command to view statistics, for example, how many total packages are installed on the system. We run 'apt-cache' followed by the command 'stats'. When we do, we see various pieces of information, such as the total number of packages installed on the system and we see how much disk space they use.

You can also use apt-cache to view dependency information. For example, let's run 'apt-cache depends gcc', and here we view the dependency information required for the gcc program. Notice that it's divided into three categories. We have two hard dependencies. These two packages have to be installed in order for gcc to run.

Then there are several suggested dependencies which are good to have, but aren't actually required, and then there's some recommended dependencies as well. If you want to just check and see whether a particular package is installed on the system or not, you can run 'apt-cache' and then enter 'pkgnames' followed by the name of the package in question, 'gcc', for example. Oops, I typed that wrong. We need a 'g' right there.

When I do, I see a list of all the packages installed on the system that have gcc somewhere in their name.

If you want to just view a list of all of the packages that are installed on the system. Just omit the package name from the command and just run 'apt-cache pkgnames', and a whole bunch of packages are displayed. These are all the packages that are currently installed on the system.

Use apt-get 6:39-9:05

Just as you can use dpkg to install new Debian packages on the system, you can also use apt to install new packages on the system, but doing so has a lot of advantages over dpkg. First of all, the apt-get command can download packages from an online repository. Meaning you don't have to go out and manually search for a package on various web pages trying to find the right one and download it.

In addition, the apt command can also resolve all of the dependencies for you. If there are other packages that the package you're trying to install depends upon, and they're not installed on your system, then the apt command will go out and get those packages for you and automatically install them. That's why the apt command is, by far and away, the preferred option for managing Debian packages.

To do this, we need to first gain root level privileges to the system. I'll use the 'sudo' command to elevate to root, and then I'll specify the command that should be run as root: 'apt-get' instead of apt-cache. As this name implies, this goes out and gets packages. Then we have to specify what we want to do.

One option is to just download. You can enter 'download' here followed by the name of the package which will cause the apt-get command to go out and get the package from a repository on the internet, download it to your system, but not install it.

If you want to go ahead and just get the package, check for dependencies, and actually install it, just type install. Let's go ahead and do that.

Let's do an 'apt-get install' followed by the name of a particular package.

Let's try one name 'celestia', hit Enter, it queries the online repository. It builds the dependency tree trying to figure out what other packages are going to be required in order for this package to be installed.

Here you can see it came up with a list of several packages that have to be installed--several that should be installed in order for the executable in this package to execute on the system--and then it gives us a summary of what is going to have to happen on the system in order for this to work.

Down here it tells us how much disk space we're going to have to use in order to do that.

In this case, we need about 70 MB of additional disk space. It asks us if we want to continue. Let's say 'yes' and all of the various packages required for this package to run are being downloaded and installed.

It'll take a minute to complete. All right, the installation process is done and now I can run it by typing 'celestia' command prompt.

That's it for this demonstration.

Summary 9:05-9:24

In this demo we talked about how to manage Debian packages on a Linux system. We first talked about using the dpkg command to view packages, install packages, and uninstall packages. Then we talked about accomplishing the same tasks using online repositories using the apt commands. We looked at apt-cache and also apt-get.
