**04 – Installing Software**

**Activities**

COMP190 – Tools and Techniques for Software Development

Dickinson College

Fall 2022

**Name:**

Over the last several classes you have learned to navigate the Unix/Linux command line interface and to use a number of the common tools and filters. In this set of activities, you will again use all of those skills and add to them some new ones that allow you to add new software to a Linux installation. Along the way you’ll learn about the PATH environment variable and the administrative privileges that are necessary to install manage a Unix/Linux system.

**Environment Variables:**

As we saw in class, Linux shells use a collection of *environment variables* to maintain information that helps them to execute programs and that can be used to customize their behavior. This section explores those variables just a little more.

1. The env command will display “the environment” (i.e. all of the environment variables). Each environment variable has a name, which by convention will typically be in all capital letters (e.g. PATH). There will be an = sign, followed by the value of the variable.

a. Run the env command. Look over the output of the env command and find the environment variable that stores the time zone for the machine (Hint: It’s the one with the value that is a location in the world). What is the name of the environment variable that holds the time zone?

b. Use an echo command to display just the value of the time zone environment variable. Paste a screenshot showing your echo command and its output below. Please do not show the full output of the env command as it will be quite long.

**The Linux/Unix PATH variable:**

2. The PATH is an important environment variable that is used on all Linux/Unix systems. We mentioned the PATH variable in class and the following questions explore the PATH environment variable a little more.

a. Describe the purpose of the PATH environment variable in a sentence or two of your own works?

b. Use an echo command as illustrated in class (and in the video above) to display the value of your PATH environment variable. Paste a screenshot of the command you used and its output below.

c. Based on your answer to part b, indicate which of the following directories are “on the path” (i.e. listed in the PATH environment variable) on your Linux system?

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  | **Directory** | **On PATH (Yes/No)?** |  |
|  | /usr/bin |  |  |
|  | /home/comp190/scripts |  |  |
|  | /usr/local/bin |  |  |
|  | /usr/local/games |  |  |
|  | /sbin |  |  |
|  |  |  |  |

**Extending the PATH:**

The value of PATH that you saw above is the default value that is set by your Linux system. Sometimes you will want to add additional directories to the PATH, so that your shell will be able to find programs or scripts that you place in that directory. For example, in HW03 you created a scripts directory within your home directory and wrote your ./countdown.sh script in that directory.

3. In this question you’ll create a new script and then you’ll extend the PATH so that you can run this script from any working directory.

a. Environment variables and the echo command can be used in shell scripts as well as on the command line. Use nano or Mousepad to create a new file in your scripts directory named myenv.sh and place the following content into it:

#!/bin/bash

echo "My home is "$HOME"."

echo "My working directory is "$PWD"."

echo "My PATH is "$PATH"."

Nothing is required here. You just need to be sure to create the script.

b. Set the permissions on the myenv.sh file so that the owner, the group and the others can all execute this script. Use the ls -l command to verify that you have set the permissions correctly. Paste a screenshot showing the commands that you used and their output below

c. Run the script that you just created to ensure that it works. Paste a screenshot of the command you used and its output below.

d. Now make your home directory your working directory. Type the command myenv.sh. What happens? Briefly explain why.

e. Use the following command to add your scripts directory to the PATH environment variable:

PATH=$PATH:/home/comp190/scripts

Use echo to check that the value of the PATH has been updated. Paste a screenshot of the commands you used and their output below.

f. Now with your home directory still as your working directory. Type the command myenv.sh. What happens now? Briefly explain why.

The command you just used to extend the PATH added /home/comp190/scripts to the path in the current terminal window only. If you were to open another terminal window or shutdown the machine and restart it the PATH would revert to its old value. It is also possible to change some configuration files in Linux so that the PATH is permanently modified. You can check out the short video *LINUX BASICS IN 3 MINS : $PATH ENVIRONMENT VARIABLE:*

* <https://www.youtube.com/watch?v=abN6bvyPRxQ> (3:00)

**The Super User:**

Because Unix/Linux is a multiuser operating system it is necessary to ensure that not every user is able to modify the system configuration. In Unix, the user named root is the *super user* and has full administrative privileges to the machine. That is root may perform any operation on any file anywhere on the system. Thus, the root user has incredible power, but like any such power, it should be used sparingly. To facilitate the use of root powers only when they are needed, Linux/Unix systems provide the sudo command. The sudo command grants you temporary administrative (i.e. root) privileges on a command-by-command basis.

To see some examples of the sudo command and to get a feel for how it is used you can watch the video *How to use the sudo command: 2-Minute Linux Tips:*

* <https://www.youtube.com/watch?v=msTXNYhFtTU> (2:27)

4. Let’s first confirm that sudo does in fact run commands as the super user (i.e. as root).

a. What does the whoami command do? The description on the man page is a good source if you do not remember.

b. Run the whoami command twice. Once normally at the command prompt as your comp190 user and once as the root user by using sudo whoami. When running the command using sudo you will be prompted for your password. Use the password comp190 that you used when logging in. Paste a screenshot of the commands you used and their output below.

c. Explain in a sentence or two why the output of your commands in part b shows that the sudo command runs other commands as the super user (i.e. as root).

6. The /etc directory on a Linux/Unix machine contains many of the configuration files that control how the system operates and who is allowed to do what. To do so, many of these files contain sensitive or system wide configuration information that normal users (i.e. not root) should not be able to read or modify (i.e. write). This question explores how sudo can be used to read and (possibly) modify these files:

a. Make /etc your working directory and use the ls -l command. Find a file for which your user (comp190) does not have read permission. Screen shot just the line of output for the file you have chosen and paste it here. Please do not paste the entire ls -l output.

b. Try to use cat to display the file that you chose in part a. Give a screenshot of your command and its output. Briefly explain why the cat command fails with the message that it does.

c. Now use sudo to run cat as the root user to display the file you chose in part a. What happens now? Briefly explain why.

**Installing Software with a Package Manager:**

You now know that the sudo command allows you to run commands with super user privileges. For example, as you just did above. Or if you were to need to edit a file that requires root privileges for writing you can simply run the editor using sudo (e.g. sudo nano). However, probably the most common use of sudo is to install software. This section shows you how to install software on a Debian Linux system using a package manager with sudo.

While many Linux distros come with a lot of good software, anytime you are doing any serious development work you are going to need to regularly install new software and update old software. This section looks at how to install new software using a *package manager*, which is the primary mechanism by which software is installed and updated in Linux.

In this section, you’ll use the command line Advanced Packaging Tool (apt) to install some software. The apt is used by Debian (and many of its derivatives, e.g. Ubuntu) as their package manager. Other distros will used different package managers (yum, apk, etc.) that do the exact same types of things, just using slightly different commands and syntax. So, the types of things you learn to do in apt can also be done in any of the other package managers as well. You’ll just have to learn their specific commands.

If you’d like to hear more about package managers and the apt before you get started, you can watch the video *Package Management Basics* from LinkedIn Learning:

* <https://www.linkedin.com/learning/linux-tips-weekly/package-management-basics> (6:59)

7. The apt relies on external sites called *package repositories*. These package repositories provide *installation* *packages (or just packages)* for all of the different software that can be installed. These packages also contain information about the dependencies that are necessary for the software to work correctly. Package repositories are updated frequently with new versions of software and new dependency information. So, the first step in using apt should always be to update it with the latest information about the packages that are available for installation. The following command updates apt with the latest information from the package repositories:

sudo apt update

Run this command and paste a screenshot of the command and its output below.

8. Our Linux system does not currently have Java installed, but it might be nice to have it. So, let’s fix that by installing Java.

a. Type the command java --version on the command line. What output is generated?

b. The command to install a new piece of software with the apt is:

sudo apt install <package>

Where <package> is the name of the installation package for the software. The package for the Java Development Kit (JDK) is named default-jdk. Use apt to install the Java JDK package. Give the full command that you used.

You will notice that when you use your command, apt will give you a lot of information about all of the additional and new software packages that are to be installed. All of these are packages contain dependencies that are required by the default-jdk package. This is one of the great thinks about using a package manager, it ensures that all of the necessary dependencies are also installed. Just imagine having to install all of those dependencies by hand before you could install java!

c. When the install has completed use the java --version command again to confirm that Java has been installed. Take a screenshot of this command and its output and paste it here.

d. Let’s check that your new JDK works by making a simple Java program, compiling it and running it as follows:

* + Use a text editor (nano or Mousepad) and enter a basic “Hello World” program:

class HelloWorld {

public static void main(String[] args) {

System.out.println("Hello comp190!");

}

}

* + Save the program into a file named HelloWorld.java.
  + Compile the program: javac HelloWorld.java
  + Run the program: java HelloWorld

Take a screenshot showing the commands that you used to compile and run the program and their output and paste it here.

**Finding Packages:**

Often the trick with using apt is knowing the name of the package that you need to install to get the software that you want. Fortunately, there are a few ways to search for packages that contain what you want.

* You can browse the packages by category on-line:
  + <https://packages.debian.org/bullseye/>
* You can search using an on-line search feature.
  + <https://www.debian.org/distrib/packages#search_packages>
* You can use your favorite search engine and search for something like:
  + Debian apt install <what I want to install>
* You can search using the apt program.
  + We’ll learn how to use this one here, because I know you can use a search engine, and this one can be quite useful.

9. The apt search <string> command will search for packages that contain <string> in their description. There is a classic Unix program called cowsay that is silly but fun (at least for a few minutes).

a. Use the apt search command to find the packages related to cowsay. Take a screenshot of the command you used and its output and paste it here.

b. In your output above the package names will be highlighted in green. What packages are related to the cowsay program?

c. Use the apt to install the package for the “graphical cowsay program”. What command did you use?

10. The apt will install the program into the /usr/games directory.

a. Check the files in that directory. What are the names of the programs that were installed?

b. The xcowsay "<string>" command will run the program where the string is what you would like the cow to say. Try to run the command: xcowsay command with a <string> of your choice. What happens? Briefly explain why. Hint: See question #3.

c. Run the xcowsay program with your <string> by using an absolute path. Give the command you used below. Cute and silly, right? ☺ Take a screenshot of the cow saying your <string> and paste it below.

d. Using absolute paths for frequently used program can be annoying, and I know you want to use xcowsay all the time… so let’s make it better.

Give a command that will add /usr/games to your PATH environment variable (Hint: See #3). Then display your PATH variable to ensure that /usr/games has been added. Give a screenshot of your commands and their outputs below. Also be sure to test that you can now run xcowsay without specifying its absolute path.

Disclaimer: The xcowsay program will only say what you tell it to say. However, that package also installs the fortune package and the xcowfortune program, which display fortune messages. The maintainers of that package have attempted to move fortunes with potentially offensive material out of the main database for the program. Their judgement may not be perfect. Please do not run the fortune program or the xcowfortune program.

**Resolving Missing Dependencies:**

If you remember back to when you ran Firefox from the command line (firefox-esr) in the previous homework there were a few error messages about missing libraries. Those are an example of missing dependencies. Missing dependencies may occur when a package doesn’t correctly list all dependencies, when a piece of software has been upgraded or when software is installed manually without a package manager. The apt can be used to install the missing dependencies.

11. In this question you will fix those errors by installing the missing dependencies for the Firefox browser. While this is exercise is specific to the Firefox browser on our machine, the idea of how to identify missing dependencies from error messages, identify the necessary packages and install them is much more general.

a. Run Firefox with the firefox-esr command. Paste screenshot of the output that appears in the Terminal window here.

b. There should be two error messages in your output in part a. Each one of these messages indicates that a particular library is missing. In Linux/Unix libraries are basically files that contain functions used by other programs. These libraries are named with a prefix of lib to indicate that they are libraries. Examine the output in part a and indicate below which two libraries are missing?

c. Use the apt search command to find the package that provides each of the missing libraries.

Most libraries have two packages associated with them. A package for those who will be developing new software using the library and those who will just be using software that depends on the library. The package for those developing new software contains source code that is needed for that development, the other package does not. The name of a package needed by those developing new software with the library will have a suffix of

-dev. The package needed to use software that depends on the library will not have the

-dev suffix. Because you will just be using firefox-esr which depends on these libraries. It is also common for the packages for libraries to include a version number (e.g. libmything1 or libstuff3).

Which two packages will you need to install?

d. Use apt to install the two libraries you identified in part c. Give the full commands that you used here.

e. Run Firefox again. If either of the error messages still appear, use apt remove to uninstall the packages you installed and revisit parts a-d to try again with different libraries. Revise until the missing dependencies have been successfully installed and the error messages no longer appear when running Firefox.

There is nothing needed here. Be sure to revise until parts a-d should resolve the issue with the missing dependencies.

**The Synaptic Package Manager:**

As you know, usually when something can be done on the CLI it can also be done using a GUI. Installing software is no different. So, it won’t be surprising that it is possible to use the apt through a GUI as well. Here you’ll use the Synaptic Package Manager to install a game of your choice, just to get a feel for how it works.

13. The GUI for the apt is called the Synaptic Package Manager.

a. Launch the Synaptic Package Manager using the command: synaptic.

What does the dialog that appears tell you? Why?

b. Quit the Synaptic Package Manager and then run it again with a command that will give it the administrative privileges that it needs in order to install new software. Give a screenshot of the command that you used in the box below.

c. Read the information in the dialog box that appears. If you don’t see a dialog or closed it, choose “Quick Introduction” from the “Help” menu. What is one way that you can mark a package for installation?

d. Browse the sections (i.e. categories) of packages (top left) and find the “Games and Amusement” category. Browse through the available game packages and find one that appeals to you. Note: The package names are not always that descriptive, but if you click on the package name, more information about the package appears below. Once you have chosen a game, mark its package for installation. Then click the “Apply” button to install it.

What game did you install and what is the name of its package?

e. All of the packages in “Games and Amusement” are installed into the /usr/games directory. Examine that directory to find the name of the program(s) that the package installed. Run the game that you installed and paste a screenshot of it running below. Note: there are a few of the “Games and Amusement” packages that do not work. If you are unlucky and have chosen one of those, go back to part d and try another.

**Summary:**

14. Complete the table below by filling in the Linux command or symbol that corresponds to each task.

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  | **Command** | **Task** |  |
|  |  | Display all of the environment variables. |  |
|  |  | Write text to standard output. |  |
|  |  | Display the PATH environment variable. |  |
|  |  | Display the absolute path to a program on the PATH. |  |
|  |  | Display the groups in which your user is a member. |  |
|  |  | Update the package information for apt. |  |
|  |  | Install new software with apt. |  |
|  |  | Remove an installed package with apt. |  |
|  |  | Search for a package using apt. |  |
|  |  | Run the Synaptic package manager. |  |
|  |  | Compile a java program. |  |
|  |  | Run a java program. |  |
|  |  |  |  |

**Optional:** To help us improve and scope these activities for future semesters please consider providing the following feedback.

a. Approximately how much time did you spend on this activity outside of class time?

b. Please comment on any particular challenges you faced in completing this activity.

**Acknowledgements:**

Some materials, questions and resources have been adapted from activities posted on foss2serve.org.

* <http://foss2serve.org/index.php/Linux_Package_Management_(Distribute_Your_App)>
* <http://foss2serve.org/index.php/Fedora_Install_Activity>