System Administration for Beginners

Week 5 Homework

March 8, 2010

1 Introduction

This homework will deal with packages installation and basic configuration. Everyone should know how to install and configure Apache using the two different methods we covered in class. That will not be the only thing to install your server, however. Unless your are working on a server that will be dedicated to only one task, it is most likely that you will have to install other packages, daemons, and other utilities to make your life as a system administrator easier, or life for users harder.

2 Homework

2.1 Server Daemon

This question is not meant to be difficult but it will test your understanding of the course material covered so far. It is meant to be a easy/fun assignment after a grueling lab and gives you a basic idea of how machines on a network communicate.

2.1.1 Submission Instructions

Please submit a history of commands, and answers to all questions (Anything that is numbered) to `sanjayk+decal@ocf.berkeley.edu`. Include your `inst` account login and server name (iXY) in your submission.

2.1.2 Java

Your first assignment is to download/install the Java Development Kit from: `http://java.sun.com/javase/downloads/index.jsp`.

NOTE: Don't be alarmed if you've never seen or learned a programming question. Most of what we ask can be easily looked up or we provide a walkthrough of what to do.

HINT: Do not forget about your `$PATH` variable. Make sure the Java files are in a directory listed in `echo $PATH`.

1. In your own words describe what the Java Development Kit does. It may help to do a bit of reading on Java and its history.

2.1.3 Digging Deeper

As you may have now figured out the Java Development Kit, is a very specific application. We will use this to demonstrate on a small scale how the internet works.

First acquire the following files, and move them to your virtual server.

```
http://tempest.ocf.berkeley.edu/users/sanjayk/decal/Client.
java
http://tempest.ocf.berkeley.edu/users/sanjayk/decal/Server.
java
```

Then, run the command: `javac Client.java && javac Server.java`

2. Describe what happened. (Hint: Look up `javac`. What does it do?)

Now, connect to your server in another terminal. Run `java Server` in one terminal and `java Client` in the other in that order.

3. What is the output? Try running the commands in a different order (i.e. `java Client`) first, what happens then?

2.1.4 Mini-Daemon

You will now take a peek into the guts of what's making this all happen and change it up a bit. The art of modifying code that is not yours is something that comes up fairly frequently when administering systems with custom written tools from long ago. Open `Client.java` in your favorite text editor and take a look at the code. It is not a simple "hello world" is it?

4. Notice the word `Socket` shows up a lot in the code. What is an internet socket?

Now, convince yourself that that this code allows for the transfer of data over TCP/IP (Remember last weeks lecture). Your task is to modify both `Server.java`, and `Client.java` to send the phrase hello world across a network. Believe it or not this simple code is essentially the framework of the internet.

Here are the basic steps (You will need to work out the details, and modify the code appropriately):

- Edit `Server.java` and `Client.java` to use port `99XY` where XY is your server name.

- Edit `Client.java` to use hostname `tsunami.ocf.berkeley.edu`.
- Copy `Server.java` over to your OCF account and log in to the machine `tsunami.ocf.berkeley.edu`.
- Compile `Server.java` on `tsunami.ocf.berkeley.edu` with the command `javac Server.java`.
- Compile `Client.java` on your virtual server with the command `javac Client.java`.
- Run the server on `tsunami.ocf.berkeley.edu`.
- Run the Client on your virtual server.

Feel free to contact one of the facilitators if at any point in the directions above you get confused (Kudos if you figure it out by looking it up yourself).

5. How long does it take for "hello world" to show up on `tsunami`? What can you say about this based on last weeks lecture?
6. What are some of the protocols used to send hello world from your virtual server to tsunami? (Hint: Think about the error messages and the phone vs. mailman analogy)
7. What are the limitations of this program?

2.2 sudo

In lecture, we discussed the concepts of regular users, root users, and superusers. By default, Debian does not come with the `sudo` package, but does come with the **APT** system that was discussed in detail this week. Remember that working as the root user is sometimes hazardous if you are unsure of what you are doing.

Install the the `sudo` package using the APT system and add an entry to the `sudoers` file that grants everyone on the `wheel` group full `sudo` access. Then add everyone to the group `wheel`. You can test this out by running `sudo -l` as a normal user.

NOTE There is a specific command you need to run to edit the `sudoers` file. Try reading the `man` page for `sudo` for more information.

2.2.1 Submission Instructions

Please turn in your homework at the beginning of class, with the assignment title, your server name, and `inst` logins at the top.

2.2.2 Questions

1. Describe the process that you took to install `sudo`, add users to the group `wheel`, and how you gave full `sudo` access to the group `wheel`.

2. What is the command syntax for `sudo`? That is, if I wanted to run a certain command that required root privileges and I only had `sudo` access, how would I do it?

3. How can I pull up a root shell using `sudo`? Essentially this would be the same as me running as root user, even though I do not have the root password.

4. I want to allow the user `nubsysadmin`, a beginning system administrator, access to a program `requiresrootprivs` that requires root privileges. What is the exact line in the `sudoers` file that I would input?

5. Let's try an actual example of the previous question. Using resources available, what is a command or program located on your server that you may want to restrict to specific users (you can install this program if it's not available on your server)? Add a new user to your system and give them `sudo` access to this program. Log in as the user you created and try starting (or stopping, if applicable) the program using `sudo`.

6. I don't remember where I placed the `sudoers` file. What command can I use to edit the `sudo` privileges of another user? Do I need to be a root user or can I use `sudo` to do this?

7. `sudo` commands are logged in a file for accountability. Where is this file located, and what does each entry in the log contain? What could someone with `sudo` privileges do to try and prevent their actions from appearing in this log file? Can you think of any way to prevent this?

8. What are the drawbacks to using `sudo`? Is it a fail-safe method to restricting access to programs? If I give full `sudo` access to a user, is that essentially giving root privileges to them?