Project 2 - Installing a LAMP Web App

Goal:

The goal of this project is to deploy the osTicket application. It's written in PHP, so you'll configure the LAMP stack for this web application. During this project I refer to the documentation for the application. It's located at on the http://osticket.com website. (http://osticket.com/wiki/Installation) You won't need it if you follow the steps in this project, however you can look at it for yourself to see how I came up with this installation process.

Instructions:

Create a Virtual Machine

First, start a command line session on your local machine. Next, move into the working folder you created for this course.

```
cd linuxclass
```

Initialize the vagrant project using the usual process of creating a directory, changing into that directory, and running "vagrant init". We'll name this vagrant project "osticket".

```
mkdir osticket
cd osticket
vagrant init jasonc/centos7
```

Configure the Virtual Machine

Edit the Vagrantfile and set the hostname of the virtual machine to "osticket". Also, assign the IP address of 10.23.45.20 to the machine.

```
config.vm.hostname = "osticket"
config.vm.network "private_network", ip: "10.23.45.20"
```

Start the Virtual Machine

Now you're ready to start the VM, test it's network connectivity, and connect to it.

```
vagrant up
```

Test the connection by pinging the virtual machine.

Window users, run the following command:

```
ping 10.23.45.20
```

Mac users, run the following command:

```
ping -c 3 10.23.45.20
```

The ping command is one simple way to test network connectivity. If you see replies, then you can safely assume the IP address is reachable and the host is up. If you see "timeout" messages then the system is not answering your ping requests. In the "real world" this doesn't necessarily mean the system is "down." It means it is not answering your ping requests which could be for a variety of reasons. However, for our purposes here if you get "timeout" messages, then you can assume this system is down or something is wrong. The first thing to try is to simply reboot the VM by running:

```
vagrant reload
```

If the ping command fails again, double check the contents of the Vagrantfile paying special attention to the config.vm.network line. Make any necessary changes, restart the virtual machine and try again. The final step is to reboot the host operating system, I.E. your physical computer. This troubleshooting step primarily applies to Windows users.

Connect to the virtual machine.

```
vagrant ssh
```

Install Apache

Start off by installing the Apache HTTP Server.

```
sudo yum install -y httpd
```

Install PHP

This web application is written in PHP, so you need to install PHP. Also, the application will be storing information in a database. This means we'll also need the php-mysql package.

```
sudo yum install -y php php-mysql
```

Install additional PHP Modules

To determine if any additional PHP modules are required for this application, look at it's documentation. Typically it will tell what is required. If it doesn't and it turns out to need additional modules, you'll need to start the web application and look at log files to determine what modules it is using and missing. Lucky for you, this application is well documented. It requires the GD, XML, MbString, and IMAP modules.

Search for each module using yum. Remember, if you need more information about a package to determine if it's the one you want, run "yum info" followed by the package name.

```
yum search php gd
yum info php-gd
sudo yum install -y php-gd

yum search php xml
sudo yum install -y php-xml

yum search php mbstring
sudo yum install -y php-mbstring

yum search php imap
```

You'll find that the PHP IMAP module doesn't exist in the default repository. Let's see if it's in EPEL.

```
sudo yum install -y epel-release
yum search php imap
```

Now you can see a package for the PHP IMAP module. Install it.

```
sudo yum install -y php-imap
```

In the future if you know all the package names for the modules you can supply them on one command, like this. Since you've already installed all the modules this command will not change anything at this point.

```
sudo yum install -y php php-mysql php-gd php-xml php-mbstring php-imap
```

Start and Enable the Web Server

Now that the web server and PHP are installed, we can go ahead and start it. We want it to be enabled on boot, so we'll enable it as well.

```
sudo systemctl start httpd
sudo systemctl enable httpd
```

Install MariaDB

Go ahead and install MariaDB. While you're at it start and enable it.

```
sudo yum install -y mariadb-server
sudo systemctl start mariadb
sudo systemctl enable mariadb
```

Secure MariaDB

Let's secure the default MariaDB installation. The only information you need to supply is the password for the root database user. For our purposes of practicing our skills we can use a simple password such as "root123". In the real world I would suggest using a stronger password. For the remaining questions you can accept the defaults by pressing ENTER.

```
sudo mysql_secure_installation
```

Example:

```
...
Enter current password for root (enter for none): (press ENTER)
Set root password? [Y/n] (press ENTER)
New password: root123
Re-enter new password: root123
Remove anonymous users? [Y/n] (press ENTER)
Disallow root login remotely? [Y/n] (press ENTER)
Remove test database and access to it? [Y/n] (press ENTER)
Reload privilege tables now? [Y/n] (press ENTER)
```

Create a Database for the Application

Use the mysqladmin command to create a database named "osticket". Enter in the root password when prompted.

```
mysqladmin -u root -p create osticket
```

Create a DB User for the Database

First, connect to the MariaDB server using the mysql client. Next, use the GRANT command to create the user, set the password, and allow full permissions to the "osticket" database. Name the user "osticket" and set the password to "osticket123". Remember to flush the privileges.

```
mysql -u root -p
> GRANT ALL on osticket.* to osticket@localhost identified by
'osticket123';
> FLUSH PRIVILEGES;
> exit
```

Download the Web Application

Create a directory to work from and change into the directory.

```
mkdir osticket
cd osticket
```

Now, download the web application. You can do that directly from your Linux system by using the `curl` command. Curl is most used to transfer data over a network such as downloading a file. The

"-L" option tells `curl` to obey redirects. Use the "-O" option causes the file to be saved locally with the same name that was used on the remote system.

By the way, when you specify multiple options to a command you can specify them separately like this: `curl -L -O`. However, when those options aren't expecting anything following them you can use a single hyphen and then combine all the single letter options like this: `curl -LO`. The order doesn't matter either, so you can do this as well: `curl -OL`. So, `curl -L -O`, `curl -O -L`, `curl -LO`, and `curl -OL` are all the same.

Remember to type in the entire command. Copying and pasting from from PDFs, web pages, and other documents may cause problems. For example, you may see a hyphen, but actually what is in the document is "pretty" character that represents that hyphen. If you were to copy and paste the contents in a terminal, it will not understand that character.

```
curl -LO http://mirror.linuxtrainingacademy.com/osticket/osTicket-v1.9.15.zip
```

Internet download location: http://osticket.com/sites/default/files/download/osTicket-v1.9.15.zip

Extract the Web Application

Now that you've downloaded the web app, it's time to extract its contents. Since it's a zip file, you can extract its contents with the `unzip` command.

```
unzip osTicket-v1.9.15.zip
```

Move the Web Application into the DocumentRoot

This particular web application included the files that should be "uploaded" to the web server in an upload directory. We're going to move them into the DocumentRoot. By default the DocumentRoot is `/var/www/html`, so that's where you'll place the files. Just like installing software with packages, putting files in most locations requires root privileges. Use sudo in conjunction with the `mv` command to perform this task. Confirm the moved with with `ls`.

```
sudo mv upload/* /var/www/html
ls -l /var/www/html
```

Per the application's documentation we need to copy the default configuration file into place and give the web server access to write to the file. The web server process runs as the apache user, so change the ownership of the file to apache so it can write to it.

```
sudo cp /var/www/html/include/ost-sampleconfig.php /var/www/html/include/ost-config.php
sudo chown apache /var/www/html/include/ost-config.php
```

Complete the Web Application Install

Open up a web browser on your local machine. Enter the IP address of the server (http://10.23.45.20) into the address bar and press enter. Follow the prompts. You'll need the database name, database username, and database password that you configured earlier. Here are some suggested inputs for the prompts:

Helpdesk Name: Linux Class Helpdesk

Default Email: support@localhost.localdomain

Admin User detail: Use your name and email address. The email address must be different from the one supplied for "Default Email". For a username, you must use something other than admin. I recommend your first name.

MySQL Table Prefix: Leave it at the default of "ost_".

MySQL Hostname: localhost

MySQL Database: osticket

MySQL User: osticket

MySQL Password: osticket123

Now that the installation is complete, the web server no longer needs write access to the configuration file. Change the owner to root.

```
sudo chown root /var/www/html/include/ost-config.php
```

Finally, the web application no longer needs its setup directory, so we'll remove it.

```
sudo rm -rf /var/www/html/setup
```

The Web App is Ready

Now you can start using the application. Feel free to explore the Staff Control Panel at http://10.23.45.20/scp/ or find out what a normal user would experience by visiting http://10.23.45.20/

Even though you can start using this system, the goal was to learn how to *deploy* the application on a Linux system. I'll leave it up to you as to how much you would like to explore the application.

OPTIONAL Configure the System to Send Outbound Emails

Since this server isn't directly connected to the Internet or to a company network with an email infrastructure, you'll need to use a third-party mail delivery system. One such system is SendGrid (http://www.sendgrid.com). As of this writing it's free to use. Visit http://www.sendgrid.com to create an account for yourself. Another option is Easy-SMTP located at http://www.easy-smtp.com. They also have a free plan that will allow to to send up to several thousand emails a month.

After you've chosen a mail delivery service, add the following configuration to the end of the `/etc/postfix/main.cf` file. It's a good idea to create a backup of a file before you edit it. Let's do that first.

```
sudo cp /etc/postfix/main.cf /etc/postfix/main.cf.orig
```

Now you're ready to make the required changed. Remember that the nano editor is an easy to use command line text editor option.

```
sudo nano /etc/postfix/main.cf
```

Paste the following into `/etc/postfix/main.cf`. Be sure to use *your* username, your password, and the hostname of the provider you're using. Save the file when you're done.

```
smtp_sasl_auth_enable = yes
smtp_sasl_password_maps = static:YOUR_USERNAME:YOUR_PASSWORD
smtp_sasl_security_options = noanonymous
smtp_tls_security_level = encrypt
header_size_limit = 4096000
relayhost = [YOUR_PROVIDERS_SMTP_HOST_NAME]:587
```

Here is an example using SendGrid:

```
smtp_sasl_auth_enable = yes
smtp_sasl_password_maps = static:testosticket:password123
smtp_sasl_security_options = noanonymous
smtp_tls_security_level = encrypt
header_size_limit = 4096000
relayhost = [smtp.sendgrid.net]:587
```

Here is an example using Easy-SMTP:

```
smtp_sasl_auth_enable = yes
smtp_sasl_password_maps = static:LINUXCLA\jason:password123
smtp_sasl_security_options = noanonymous
smtp_tls_security_level = encrypt
header_size_limit = 4096000
relayhost = [ssrs.reachmail.net]:587
```

Finally, restart the postfix service so it can load this new configuration.

```
sudo systemctl restart postfix.service
```

By the way, mail logs are located at `/var/log/maillog`.