Project - phpMyAdmin

Goal:

The goal of this project is to install and configure phpMyAdmin. It is a web application designed for managing MySQL and MariaDB databases. Once you have phpMyAdmin installed you will be able to easily create, read, update, and delete databases as well as database users.

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 "phpmyadmin".

```
mkdir phpmyadmin
cd phpmyadmin
vagrant init jasonc/centos8
```

Configure the Virtual Machine

Edit the Vagrantfile and set the hostname of the virtual machine to "phpmyadmin". Also, assign the IP address of 10.23.45.35 to the machine.

```
config.vm.hostname = "phpmyadmin"
config.vm.network "private_network", ip: "10.23.45.35"
```

Start the Virtual Machine

Now you're ready to start the VM and connect to it.

```
vagrant up
vagrant ssh
```

Install Apache, PHP, and Mariadb

Let's install the major components of the LAMP stack. First, we'll install the Apache HTTP Server.

```
sudo dnf install -y httpd
```

Next, install PHP and the required PHP modules. Because phpMyAdmin will be connecting to MariaDB, a drop-in replacement for MySQL, we'll need to install the PHP MySQL module. Also, phpMyAdmin uses JSON, so we'll need to install the PHP JSON module. Next, phpMyAdmin supports the importing of compressed databases in zip format, so we'll need to install the PHP Zip module. Finally, we need to add multi-byte string to PHP because you may work with databases that contain UTF-8 characters.

```
sudo dnf install -y php php-mysqlnd php-json php-pecl-zip php-mbstring
```

Now we can start and enable the web server.

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

Next, install the database server.

```
sudo dnf install -y mariadb-server
```

Start and enabled the database server.

```
sudo systemctl start mariadb
sudo systemctl enable mariadb
```

Secure the default MariaDB installation by running the `mysql_secure_installation` script.

```
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)
```

Install phpMyAdmin

Download phpMyAdmin. Make sure you type the following command exactly as shown. Capitalization matters!

```
curl -LO http://mirror.linuxtrainingacademy.com/phpMyAdmin/phpMyAdmin-5.0.2-all-languages.zip
```

Internet download location:
https://files.phpmyadmin.net/phpMyAdmin/5.0.2/phpMyAdmin-5.0.2-all-languages.zip

Extract the contents of the archive file with the `unzip` command.

```
unzip phpMyAdmin-5.0.2-all-languages.zip
```

NOTE: If you get an error when unzipping the file, it is probably due to a typing mistake when downloading the file. If you have a typing mistake, then the zip file will contain an error message instead of the actual zip file. To check this, run this command:

```
file phpMyAdmin-5.0.2-all-languages.zip
```

If "HTML document, ASCII text" is returned, delete the file and download it again, checking your typing very carefully. If "Zip archive data" is returned, then the file downloaded successfully.

```
rm phpMyAdmin-5.0.2-all-languages.zip
```

Move the phpMyAdmin application into the DocumentRoot of the web server. Confirm the move with `ls`.

```
sudo mv phpMyAdmin-5.0.2-all-languages/* /var/www/html/
ls -l /var/www/html
```

Configure phpMyAdmin

The configuration of phpMyAdmin is controlled by a file named `config.inc.php`. It comes with a sample configuration file, so lets use that as the basis of our configuration. Copy the sample configuration file to the name of the actual configuration file we'll use, `config.inc.php`.

```
sudo cp /var/www/html/config.sample.inc.php /var/www/html/config.inc.php
```

There is only one mandatory setting that we much change. We need to generate a string of characters that will be used to encrypt the data stored in cookies by the web application. You can think of this like a password, but you will never have to actively use the password. It will simply be used as a key to generate encrypted data. This string should be at least 32 characters long, but it is fine if it is longer.

One way to get a string of that size is by using the `md5sum` command. The md5sum command generates a compact digital fingerprint consisting of 32 characters. We can use the `date` command as input to the `md5sum` command to generate a string.

```
date | md5sum
```

NOTE: The md5sum tool may not be available in future releases of the major Linux distributions. You can also use other hashing tools such as `shasum`, `sha256sum`, and `sha512sum`. You could also use a random password generator. Many are freely available on the Internet.

Open the configuration file for editing.

```
sudo nano /var/www/html/config.inc.php
```

Find this line of configuration:

```
$cfg['blowfish_secret'] = '';
```

Insert the string between the two single quotes like so:

```
$cfg['blowfish_secret'] = '12345678901234567890123456789012';
```

Next, phpMyAdmin needs a place to store temporary files. Let's create a `tmp` directory and change the ownership of the directory to the user that is running the web server, apache. This way the web application can create, read, update, and delete files from that directory.

```
sudo mkdir /var/www/html/tmp
sudo chown apache /var/www/html/tmp
```

Start Using phpMyAdmin

Open up a web browser on your local system and visit http://10.23.45.35/. You should see a login page.

You have to log in with a valid MariaDB user. By default, phpMyAdmin requires that the root user have a password before it allows you to login as the root user. You either need to make sure that

root database user has a password -- remember, you should run `mysql_secure_installation` after you install MariaDB -- or log in as another database user. If you log in as a non-root user, you will only be able to access the resources you would normally have access to as that database user no matter how you connected to MariaDB.

Create a User and a Database

For now, lets login as the database root user with the password you configured earlier. One of the most common database administration tasks you'll have to perform is creating a database for an application, creating a user that can access just that database, and setting a password for that user. You can do all three of these things with just a couple of clicks.

First click on the "User accounts" tab. Next click "Add user account". From here you fill out a simple form. You specify a name for the user, where they can access the database from (typically localhost), and assign them a password or have phpMyAdmin generate a password for you.

You can also create a database at the same time by checking a couple of boxes. Make sure you check the box beside "Create database with same name and grant all privileges".

Here is some suggested data for you to practice with:

> User name: webapp01
> Host name: localhost (Select "Local" in the drop-down box.)
> Password: webapp01pass123
> Re-type: webapp01pass123
> Create database with same name and grant all privileges: Click the check box.

When you submit this form by clicking the "Go" button at the bottom of the page, the user and database will be created. You can confirm this by looking at the user accounts tab and the databases tab.

Click on the "User accounts" table to reload the list of database users. Confirm you see the webapp01 user.

Next, click on the "Databases" tab and confirm you see the webapp01 database.

Explore phpMyAdmin

Feel free to explore phpMyAdmin. It's a very powerful database management tool. You can execute SQL statements directly against a database, backup databases (export), and more. If your primary focus is being a Linux Administrator and not a Database Administrator (DBA), then using a tool like phpMyAdmin can be the way to go.