Lab VI – From SQL Injection Attacks to Shell

CPS 499-02/592-02

Software/Language Based Security

Fall 2020

Dr. Phu Phung

Evan Krimpenfort

Part I: Exploiting the SQL Injection Vulnerabilities to obtain data from the database

Task I: Discover the SQL vulnerabilities

a. Display UD email ID on the website

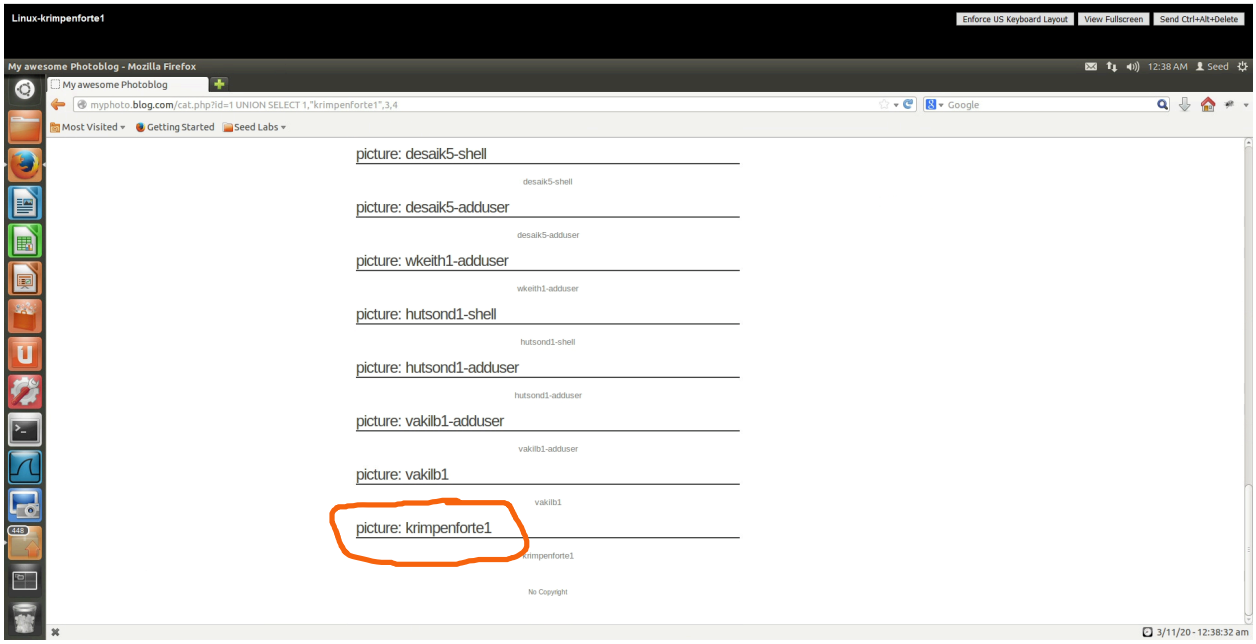

Figure 1: putting in my email ID for column 2.

b. Find current_user()

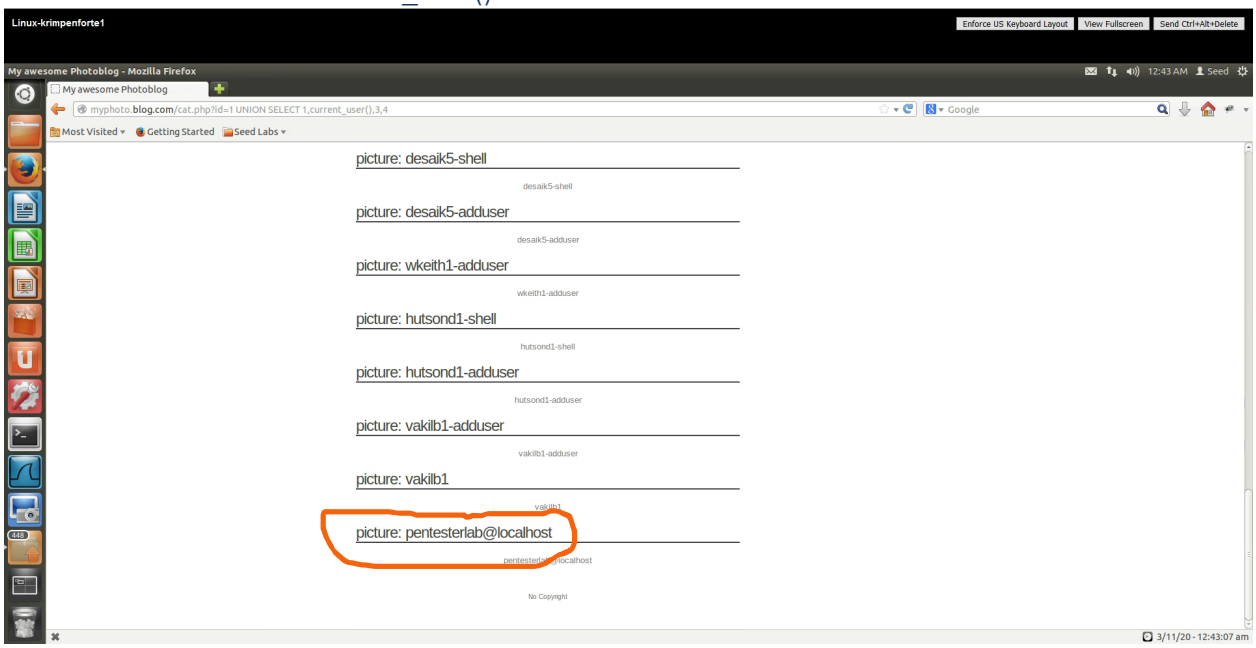

Figure 2: putting in the function "current_user()"

c. Find database()

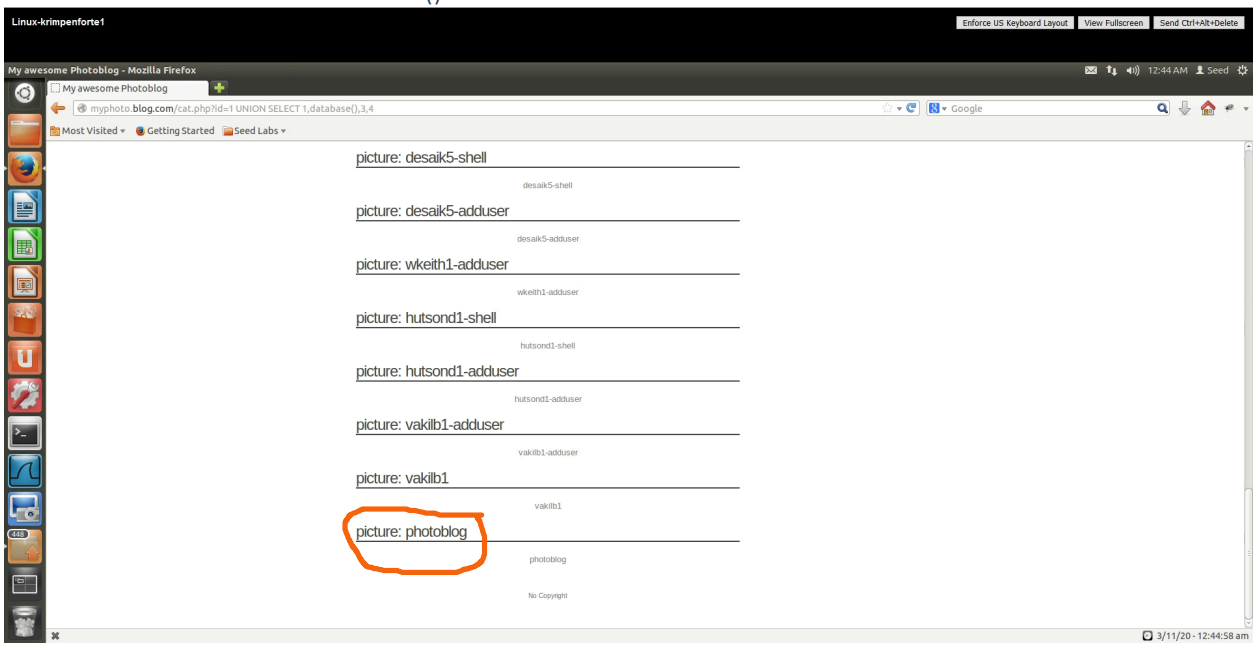

Figure 3: putting in the function "database()"

d. Display all of the information together

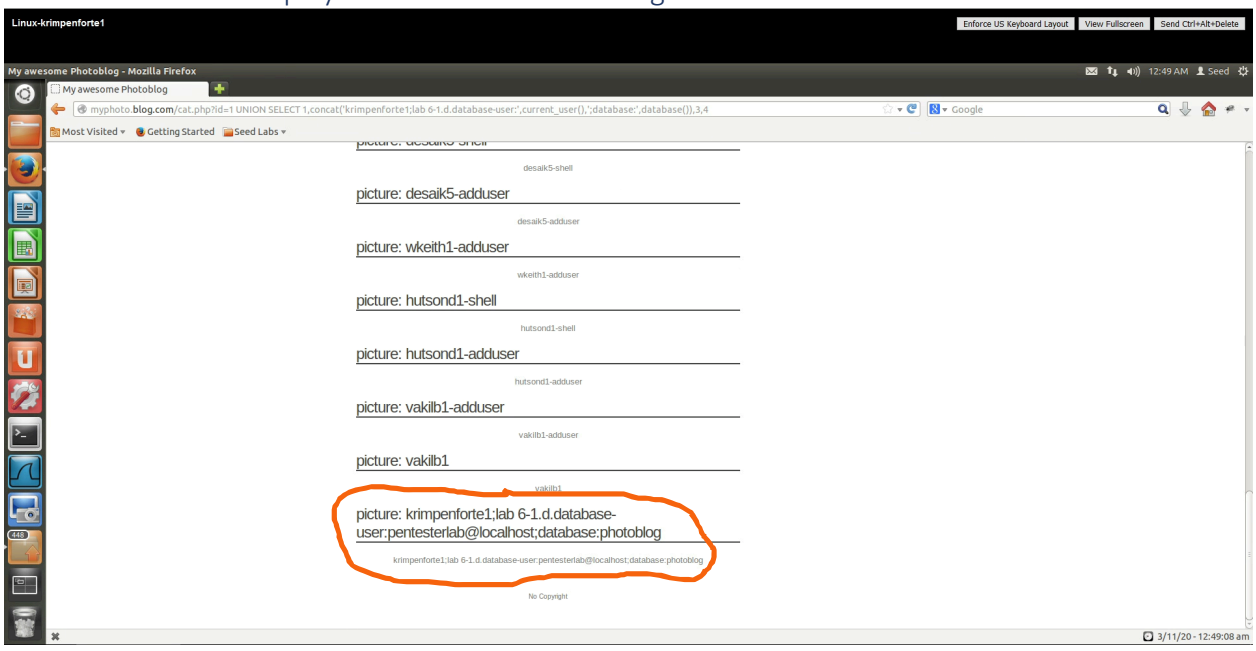

Figure 4: putting everything together using "concat()"

Task II: Retrieve the data from the database

a. Display all tables with their columns

i. Injected query results

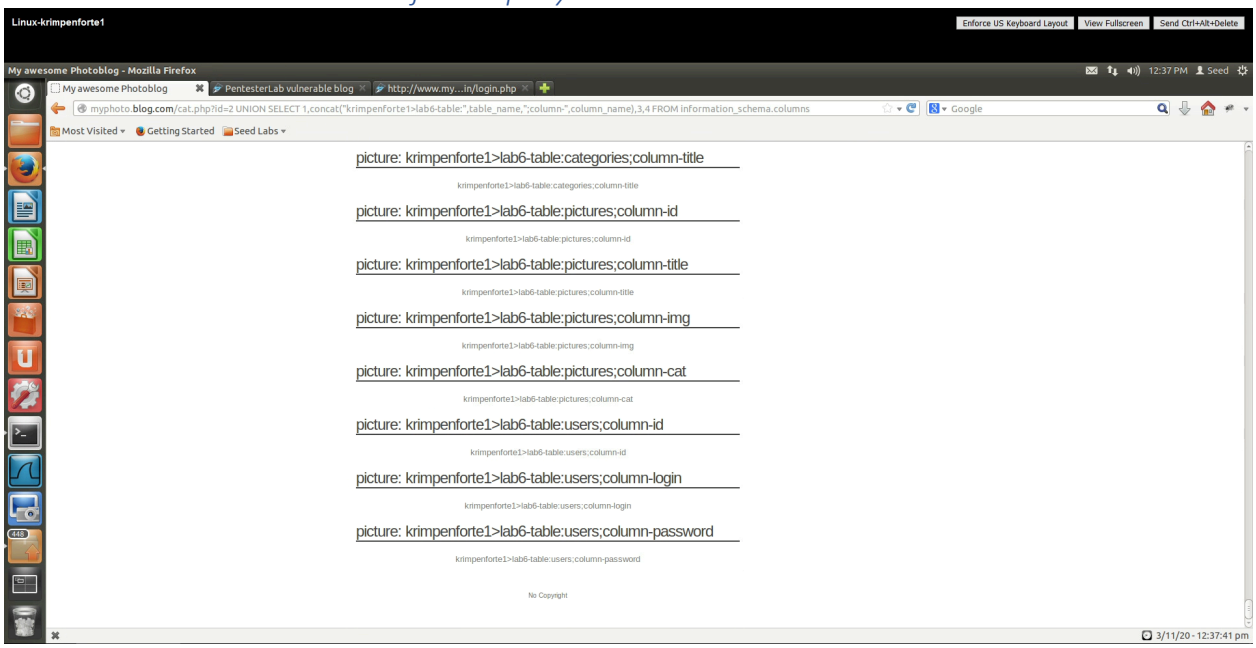

Figure 5: All tables and all of their respective columns

ii. Table Name and its columns

From looking at Figure 5, the table name that holds the data we want is called *users* and the columns that are inside of *users* that we want are called *login* and *password*.

b. Display contents of the table

iii. Injected query results

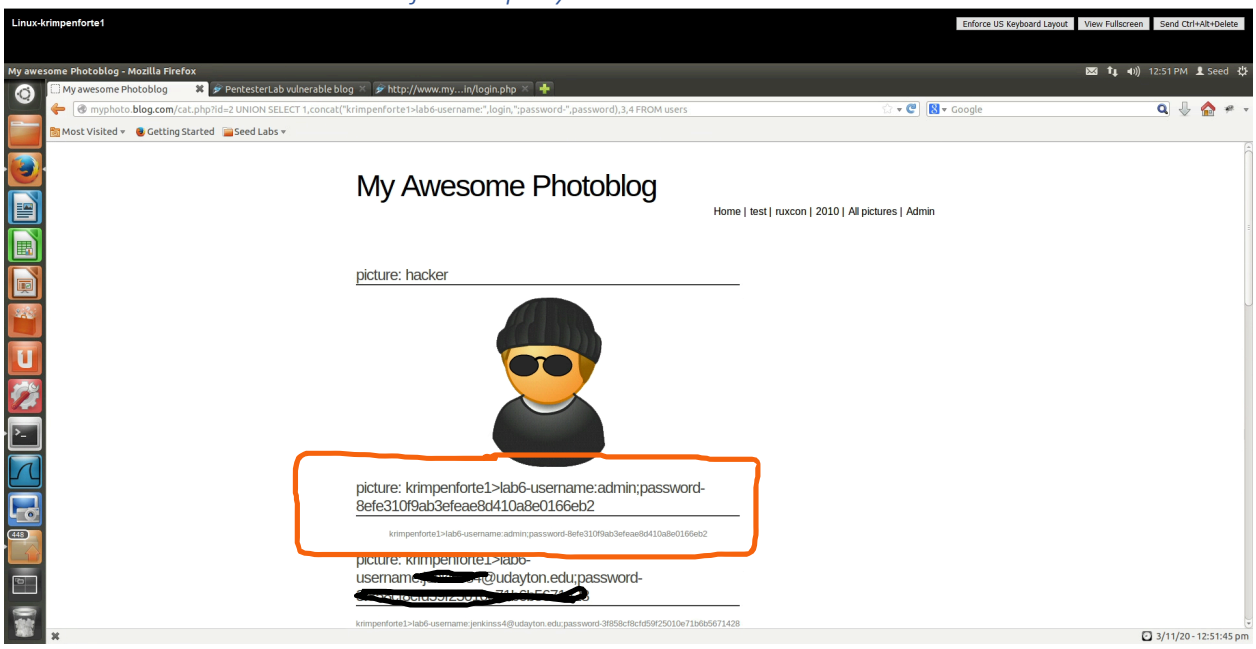

Figure 6: retrieving the admin username and password

iv. Username and Password

Figure 7: admin and its password

c. Log into the system with the made discovery

v. Demonstration

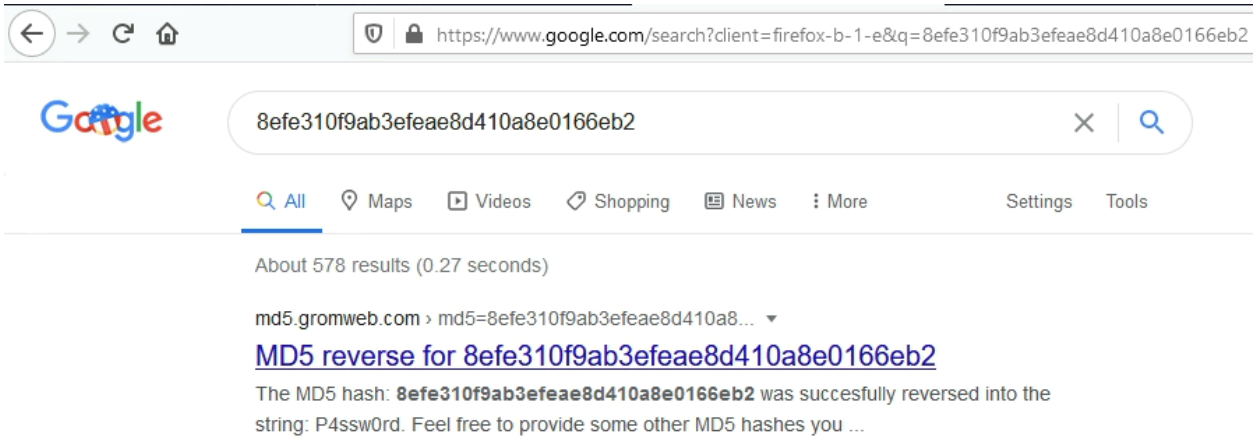

Figure 8: Looking up the password

To find the plaintext value, I searched the hash code up in Google and a website says the string is “P4ssw0rd.”

vi. Results

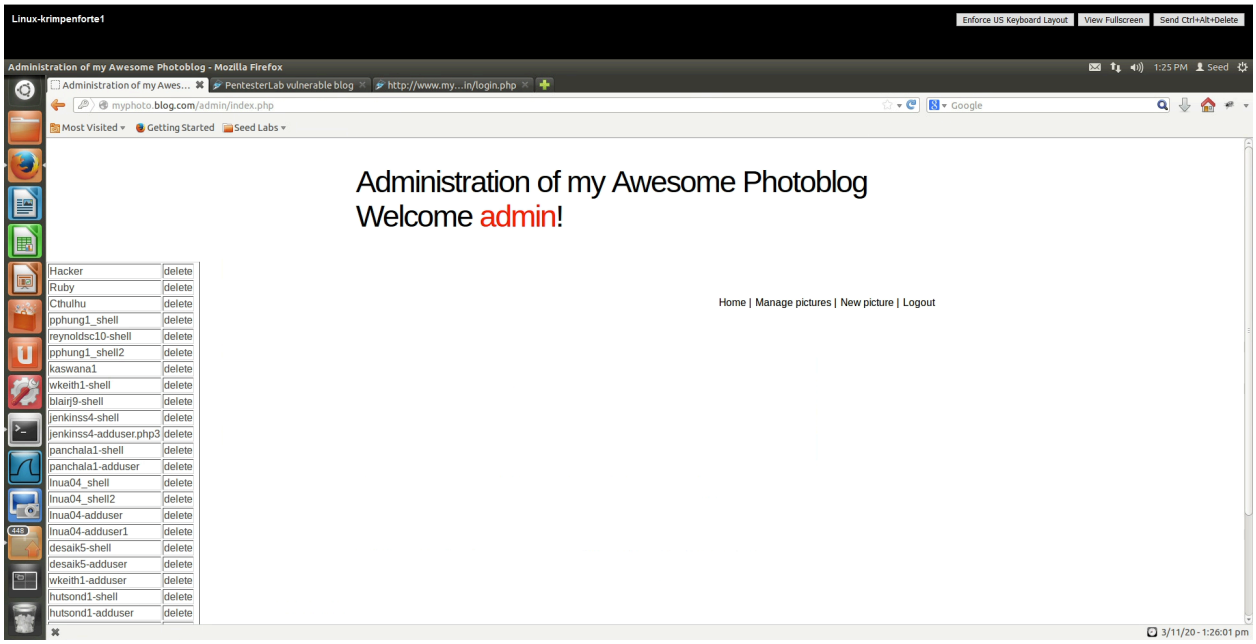

Figure 9: Logged into admin using the username “admin” and password “P4ssw0rd”

Part II: Access to the web application and modify the system

Task III: Upload a php application to execute commands

a. Uploading a new file

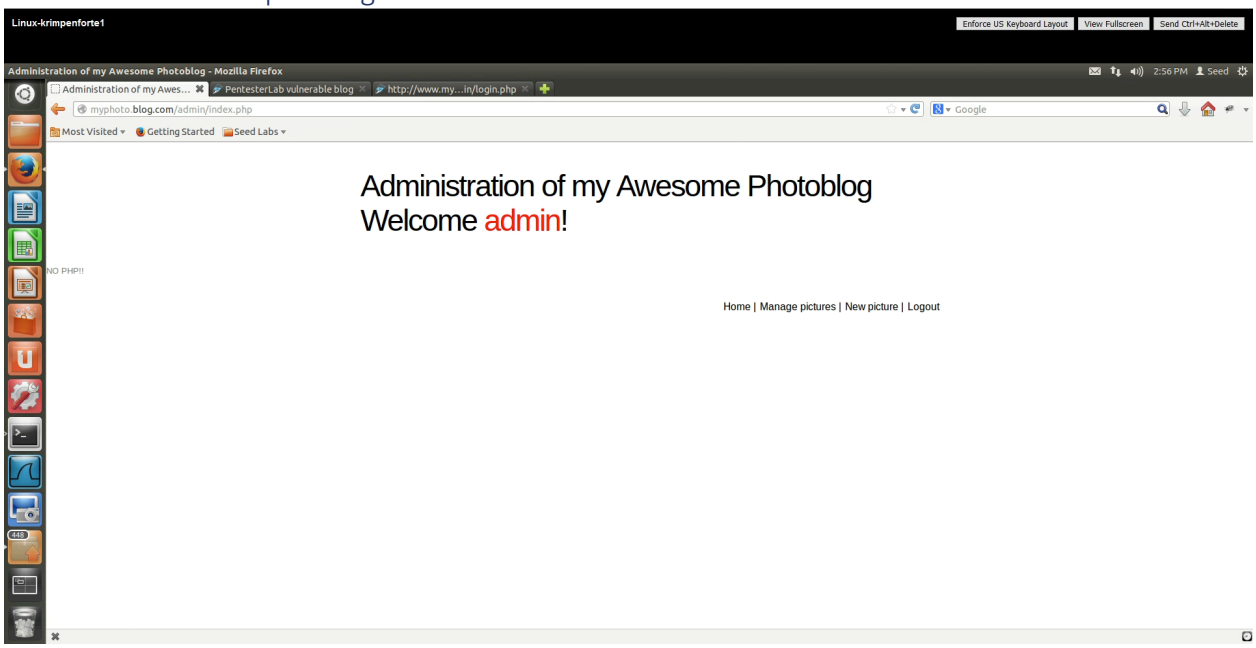

Figure 10: When trying to add the php file, there was an error

This error is caused above because the website had some sort of input validation. It checked to see if the code was a php file extension and it was.

b. Try again…

i. Demonstration

Figure 11: successfully added a php file with the extension php3

ii. Interaction

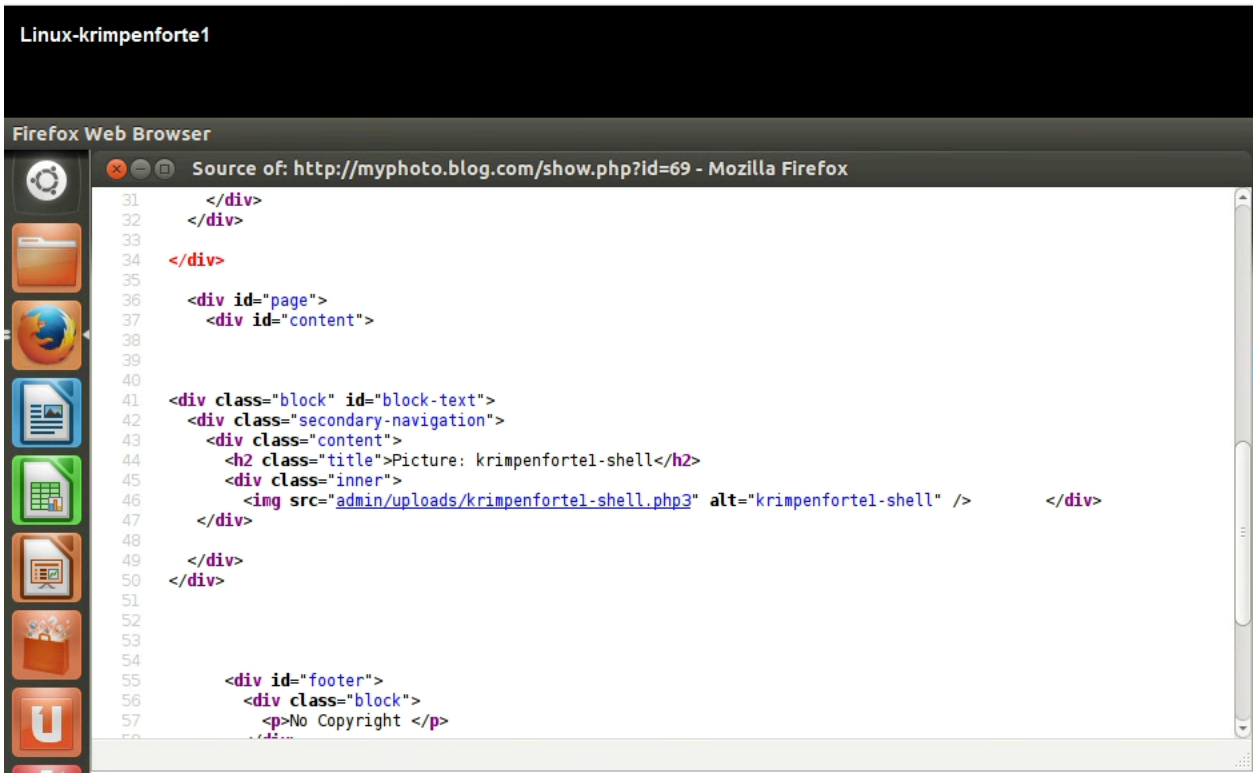

Figure 12: Where the code is

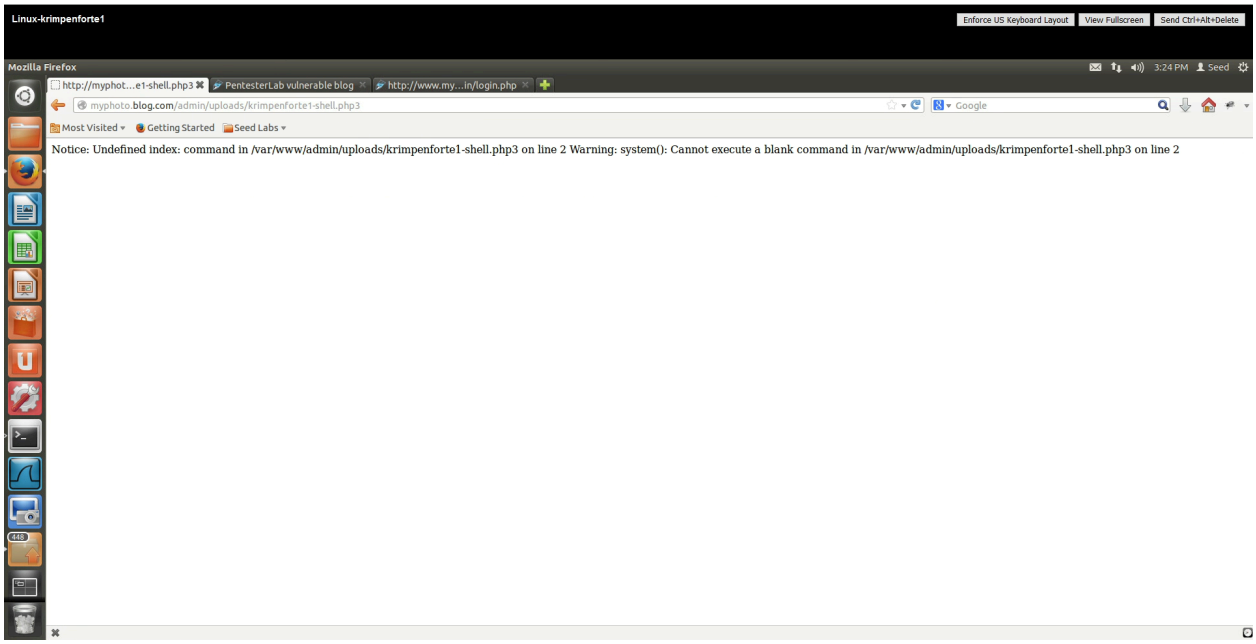

Figure 13: when ran straight, there is an issue because "command" isn't used

There is an issue here because the placeholder called "command" isn't instantiated with a command to process in *$_GET*. Therefore, in the next step, we will use command in the URL.

iii. Execution

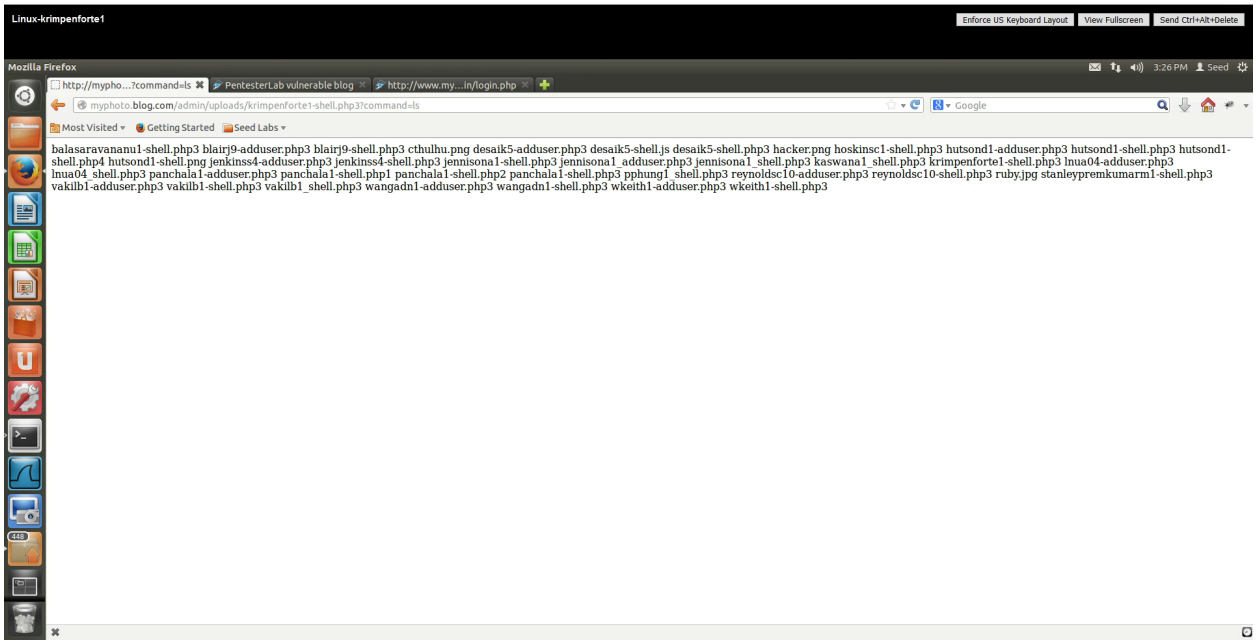

Figure 14: ?command=ls

The command entered was ls and now the code works.

Task IV: Execute System commands to exploit the system further

a. Repeat with new functions

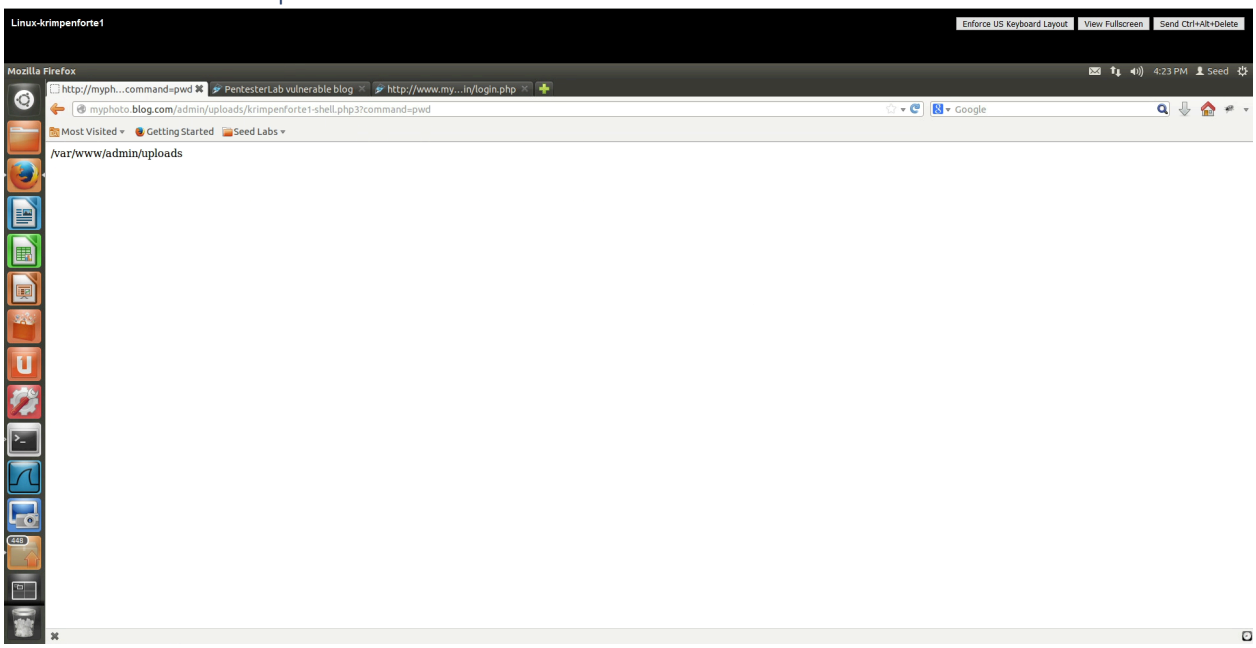

Figure 15: ?command=pwd

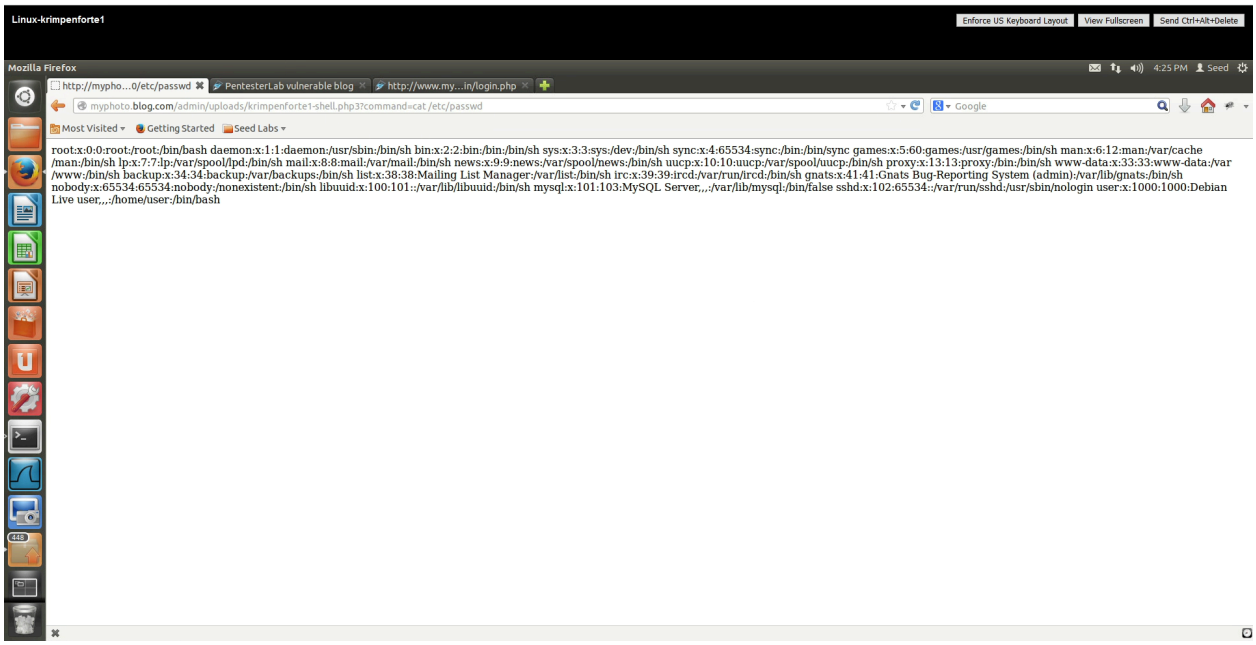

Figure 16: ?command=cat /etc/passwd

Because we can see /etc/passwd, we have root access with the admin login.

b. Similar tests

i. List all files and folders

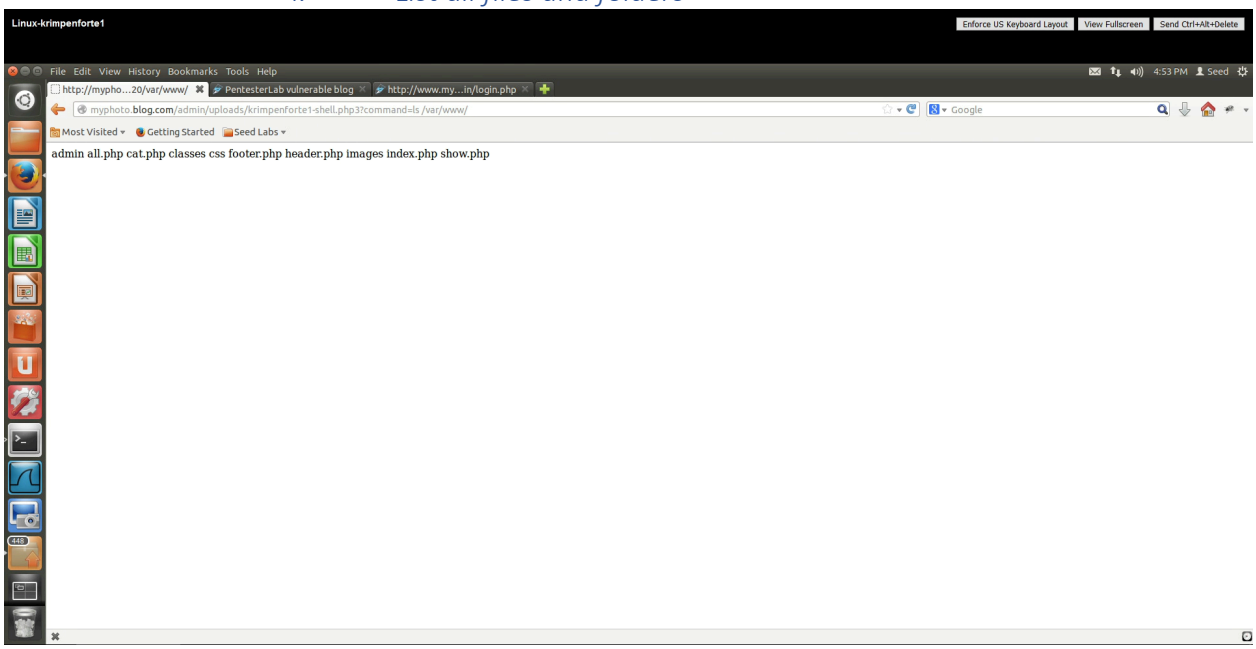

Figure 17: ?command=ls /var/www/

The shell call of “pwd” said that the root directory is /var/www/.

ii. View cat /var/www/classes/db.php

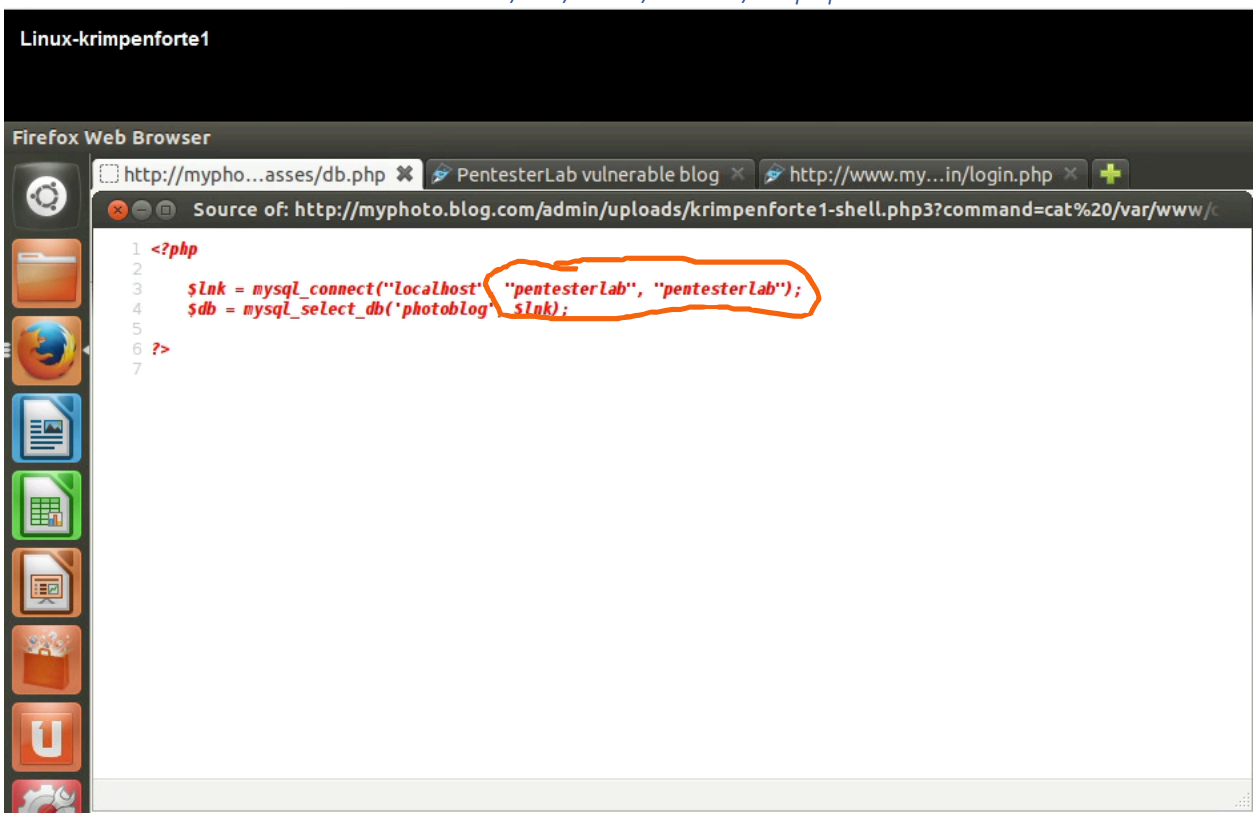

Figure 18: Looking at the source code of the db.php file

iii. Figure out the credentials

As seen in Figure 18, the username and password seen in this file is “pentesterlab.”

Task V: Insert new data into the database

a. Content with the filename

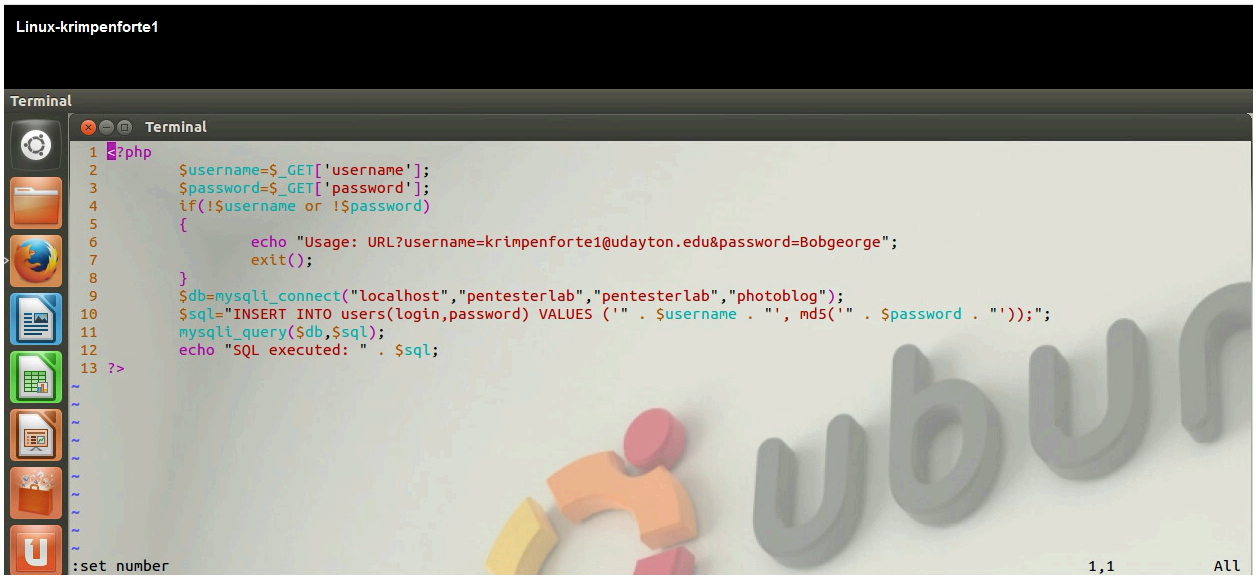

Figure 19: Code for krimpenforte1-adduser.php3

b. Upload

Figure 20: Added my code as a picture for the adduser

Was able to get another PHP file in because of the php3 extension.

c. Provide Username and Password

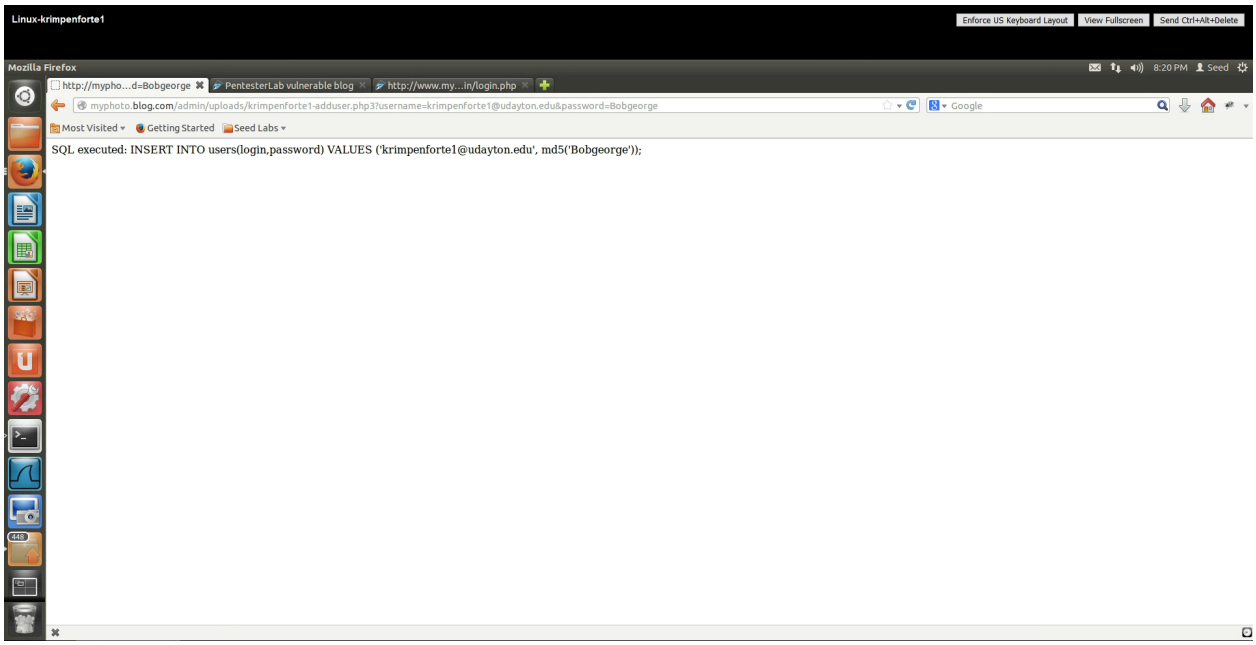

Figure 21: Success addition of an added user (me)

d. Log in with new username and password

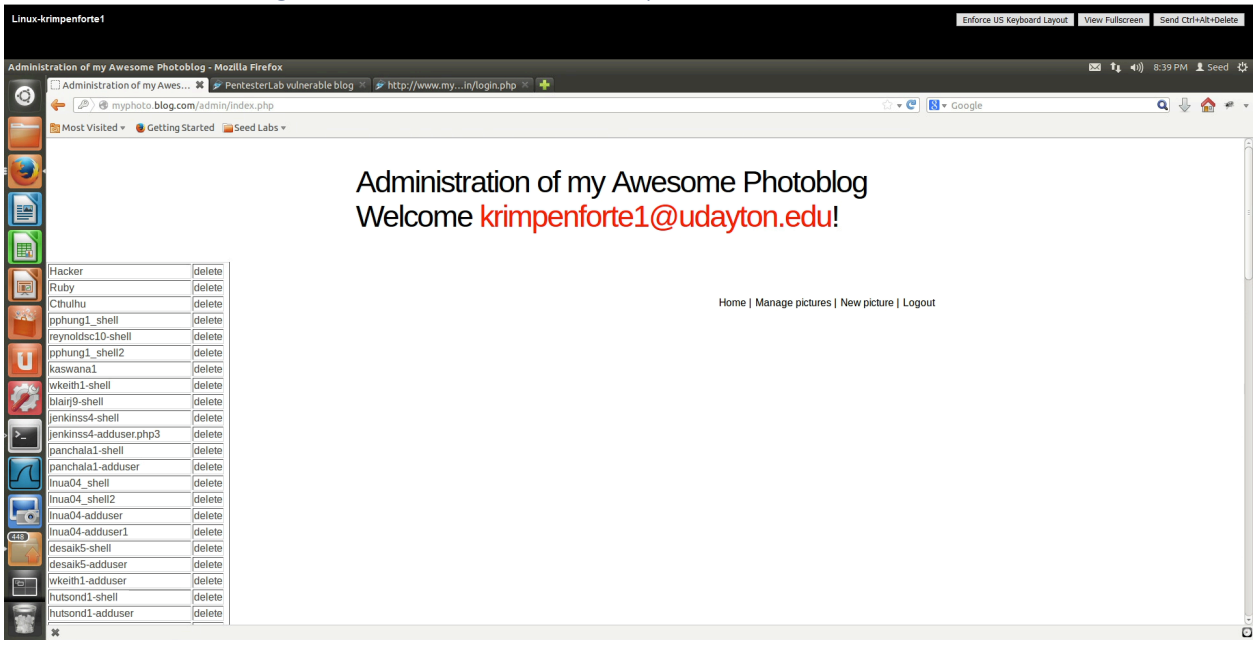

Figure 22: Successfully logged into the admin using my added username