
esercizio 14\1\25

consegna

Esercizio del Giorno Esercizio Traccia Argomento: Sfruttamento delle Vulnerabilità XSS e SQL Injection sulla DVWA

Obiettivi: Configurare il laboratorio virtuale per sfruttare con successo le vulnerabilità XSS e SQL Injection sulla Damn Vulnerable Web Application DVWA.

Istruzioni per l'Esercizio:

Configurazione del Laboratorio:

- Configurate il vostro ambiente virtuale in modo che la macchina DVWA sia raggiungibile dalla macchina Kali Linux (l'attaccante).
- Verificate la comunicazione tra le due macchine utilizzando il comando ping. Impostazione della DVWA
- Accedete alla DVWA dalla macchina Kali Linux tramite il browser.
- Navigate fino alla pagina di configurazione e settate il livello di sicurezza a LOW. Sfruttamento delle Vulnerabilità:
- Scegliete una vulnerabilità XSS reflected e una vulnerabilità SQL Injection (non blind).

SVOLGIMENTO

per cominciare andremo a settare gli iIP e le netmask delle nostre due reti interne


kali : ip = **192.168.10.2**

meta: ip =**192.168.10.1**

```
PS> ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.10.2 netmask 255.255.255.0 broadcast 192.168.10.255
    inet6 fe80::2b3e:921a:2485:2557 prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:ad:25:87 txqueuelen 1000 (Ethernet)
    RX packets 1006 bytes 201615 (284.7 KiB)
```

effettuiamo con successo una richiesta di ping da entrambe le macchine per verificare che sia tutto regolarmente impostato, poi accediamo a **dvwa** e settiamo la sicurezza su **LOW**

```
(kali@kali)-[/home/kali/Desktop]
PS> ping 192.168.10.1
PING 192.168.10.1 (192.168.10.1) 56(84) bytes of data.
64 bytes from 192.168.10.1: icmp_seq=1 ttl=64 time=2.49 ms
64 bytes from 192.168.10.1: icmp_seq=2 ttl=64 time=0.901 ms
64 bytes from 192.168.10.1: icmp_seq=3 ttl=64 time=0.193 ms
64 bytes from 192.168.10.1: icmp_seq=4 ttl=64 time=0.238 ms
```



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

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 hits/tips for each vulnerability and for each security level on their respective page.

Username: admin
Security Level: low
PHPIDS: disabled

DVWA Security

Script Security

Security Level is currently **low**.

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

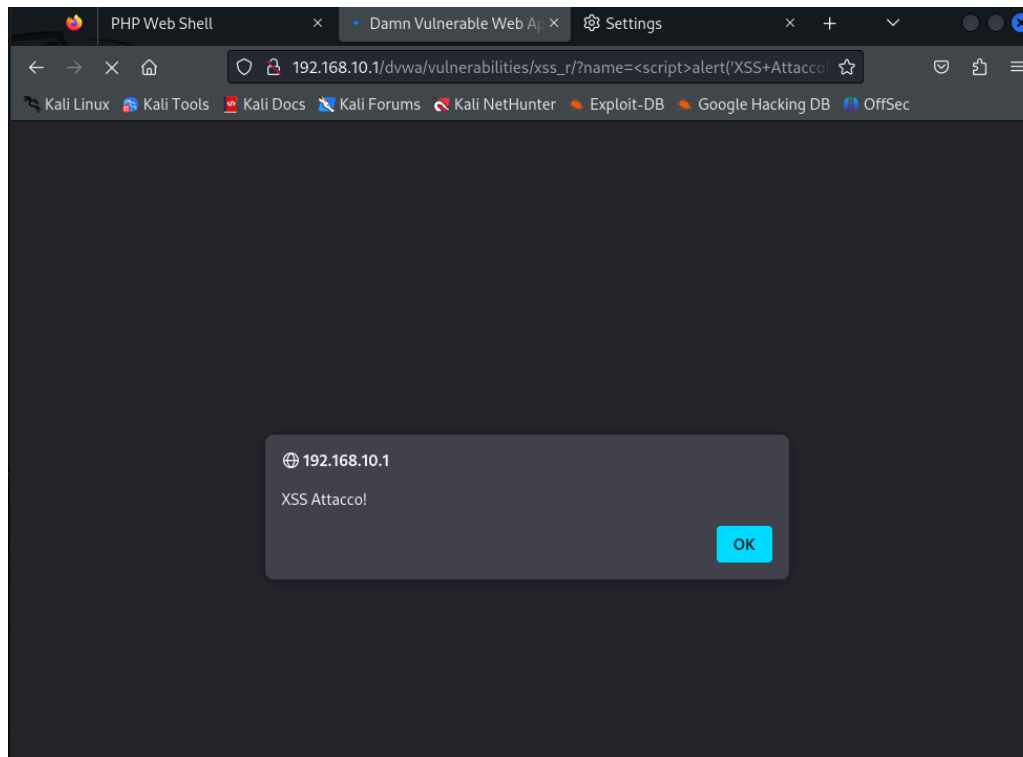
The security level changes the vulnerability level of DVWA.

low

1) XSS

procediamo andando sulla sezione **XSS reflected** e proviamo ad inviare un input malevolo, che verrà eseguito dal browser. in questo caso useremo un semplice '**ALERT**' al fine di verificare soltanto il corretto funzionamento contenente la scritta 'XSS attacco!'

```
<script>alert('XSS Attacco!');</script>
```



ecco che visualizziamo l'alert, il browser ha infatti eseguito il nostro comando tramite l'attacco XSS.

2) SQL INJECTION

Spostiamoci quindi ora alla sezione SQL injection e proviamo ad effettuare l'iniezione di codice. utilizzeremo semplicemente il valore booleano **'true'**, rappresentato in questo caso dal codice `'1 = 1'`

Vulnerability: SQL Injection (Blind)

User ID:

ID: 1 = 1
First name: admin
Surname: admin

come possiamo vedere ci ha restituito dati come

First name = admin

Surname = admin

L'accesso ai dati appena visualizzati significa che la vulnerabilità è stata sfruttata e l'attacco è riuscito.

P.S. nota a margine, nell'ultima immagine sql injection è BLIND, ma ho già provato a riefettuare il testo senza blind e il risultato è lo stesso.