



GRADO EN INGENIERÍA INFORMÁTICA

SEGURIDAD EN LOS SISTEMAS INFORMÁTICOS

DEPARTAMENTO DE INGENIERÍA INFORMÁTICA

Práctica 6: Explotación de aplicaciones web y bases de datos

Autor:

Juan Boubeta Puig, Jesús
Lagares Galán, Pedro José
Navas Pérez y Kevin J.
Valle Gómez

Fecha:

28 de noviembre de 2024

Índice

1. Objetivo	3
2. OWASP	3
3. OWASP Top 10 - 2021	4
4. OWASP Juice Shop	5
4.1. Conexión	5
5. Burp	6
5.1. Proxy web	7
5.2. Instalación de Burp	8
5.3. Uso de Burp	10
6. Ejercicios	12
6.1. Ejercicio 1: OWASP Top 10 - 2021	12
6.2. Ejercicio 2: Explotación de OWASP Juice Shop	12
6.2.1. Injection	12
6.2.2. Broken Authentication	15
6.2.3. Sensitive Data Exposure	21

Índice de figuras

1.	Captura de pantalla de OWASP Juice Shop	6
2.	Relación de las herramientas de Burp entre sí	7
3.	Salida brindada por el comando burpsuite -help	9
4.	Opciones para descargar Burpsuite Community Edition para Linux 64 bits	9
5.	Instalador de la herramienta Burp	9
6.	Creamos un proyecto temporal en Burpsuite	10
7.	Dejamos la opción por defecto respecto al fichero de configuración . .	11
8.	Abrimos el navegador que Burp pone a nuestra disposición	11
9.	Comprobamos que la opción de <i>intercept</i> está en <i>on</i>	11
10.	Vemos cómo Burp intercepta nuestra petición	12
11.	Vemos la IP de la máquina	14
12.	Hacemos clic en el icono de FoxyProxy	14
13.	Añadimos los datos para la conexión del Proxy de Burp	14
14.	Cambiamos nuestro proxy para utilizar Burp	14
15.	Introducimos datos en la pestaña de <i>login</i>	15
16.	Aparecen los datos de nuestra petición al pulsar el botón <i>forward</i> . .	15
17.	Ingresamos una tautología en el campo del usuario	16
18.	Observamos cómo nos devuelve el <i>email</i> del administrador	16
19.	Hemos conseguido acceso a la cuenta del administrador	17
20.	Localizamos la sección de Payload Options	18
21.	Seleccionamos la lista best1050	19
22.	Colocamos los datos del target	19
23.	Modificamos el <i>password</i> para utilizar la lista de contraseñas	20
24.	Coincidencia encontrada en la línea 117 de nuestro archivo (admin123)	20
25.	Comprobamos que el enlace nos llevaría hacia una dirección FTP . .	21

1. Objetivo

El objetivo principal de esta práctica es aprender y conocer el funcionamiento y la dimensión de las distintas posibles vulnerabilidades presentes en una aplicación web y su base de datos.

Cabe destacar que las páginas web y las aplicaciones web no han dejado de aumentar en cantidad y relevancia en los últimos años. Muchas veces, en la creación de estas se cometen errores, o se pasan cosas por alto, que dan lugar a vulnerabilidades en el sistema. Denominamos vulnerabilidad a una debilidad que puede ser explotada por un ataque cibernético, por ejemplo, haciendo uso de un formulario dentro de una página web, un usuario ilegítimo podría acceder a información no pública registrada en una base de datos.

Además de las vulnerabilidades a estudiar, se presentarán una serie de ejercicios y una actividad de gamificación para reforzar todo lo aprendido. No obstante, no podemos explotar estas vulnerabilidades para su aprendizaje en un entorno real, ya que esto es ilegal. Por este motivo utilizaremos la aplicación web más insegura creada para el aprendizaje conocida como **OWASP Juice Shop** [5], proporcionada por el Proyecto Abierto de Seguridad en Aplicaciones Web (OWASP) [4]. Gracias a esto podremos realizar experimentos y recrear distintos mecanismos de ataque sin que esto ponga en peligro nuestros entornos reales.

2. OWASP

OWASP es una comunidad abierta dedicada a permitir que las organizaciones desarrollen, adquieran y mantengan aplicaciones y APIs (*Application Programming Interfaces*) en las que se puedan confiar. En OWASP podemos encontrar de forma abierta y gratuita:

- Herramientas y estándares de seguridad en aplicaciones.
- Libros completos de revisiones de seguridad en aplicaciones, desarrollo de código fuente seguro y revisiones de seguridad en código fuente.
- Controles de seguridad estándar y bibliotecas.
- Presentaciones y vídeos.
- Hojas de trucos en varios temas comunes.
- Investigaciones de vanguardia.
- Capítulos locales en todo el mundo.

- Numerosas conferencias alrededor del mundo.
- Listas de correo.

Gracias a las numerosas investigaciones y revisiones de seguridad que este proyecto realiza, podemos consultar una lista con las 10 vulnerabilidades web más importantes, tal y como se describe a continuación.

3. OWASP Top 10 - 2021

Cada 4 años, OWASP publica un documento recogiendo las 10 vulnerabilidades de seguridad más importantes.

En particular, las vulnerabilidades que aparecen en el *Top Ten 2021* [6] son las siguientes:

1. ***Broken Access Control***. Esta vulnerabilidad hace que el control de acceso falle o quede inutilizado, por lo que los usuarios pueden actuar fuera de los permisos concedidos previamente. Por tanto, un usuario común podría acceder a funciones de administrador.
2. ***Cryptographic Failures***. Una de las vulnerabilidades más antiguas que existen. Consiste en un mal uso de la criptografía para proteger la información. Esta puede presentarse de numerosas formas, por ejemplo, transmitiendo contraseñas como texto plano.
3. ***Injection***. Esta vulnerabilidad consiste en la inyección de código (SQL, NoSQL, LDAP, etc.) por la falta de validación de entradas, por ejemplo, en los formularios.
4. ***Insecure Design***. Esta vulnerabilidad apareció como una categoría completamente nueva en este Top 10 - 2021. Es una categoría amplia que agrupa todas aquellas debilidades correspondientes a un diseño falto de control o ineficaz. Por ejemplo, si durante el diseño de nuestra aplicación no integramos pruebas unitarias, podemos estar pasando por alto problemas de seguridad de nuestra futura aplicación.
5. ***Security Misconfiguration***. Una aplicación puede hacerse vulnerable porque haya fallas en su creación y, además, porque se configure de forma incorrecta. Esta vulnerabilidad hace alusión a esa última tarea. Por ejemplo, podemos tener una vulnerabilidad si instalamos componentes innecesarios o exponemos puertos para nuestra aplicación que no deberían ser abiertos por falta de uso.
6. ***Vulnerable and Outdated Components***. Esta vulnerabilidad consiste en

disponer de componentes de la aplicación sin la última versión, lo que genera posibles vulnerabilidades (ya que muchos parches de las aplicaciones vienen dados para solucionar fallas de seguridad). También podría darse por no actualizar componentes o no examinar regularmente el sistema en busca de errores o amenazas.

7. ***Identification and Authentication Failures***. Esta vulnerabilidad consiste en el fallo de una identificación o autenticación de usuario. Podemos encontrarla, por ejemplo, en una aplicación que permita realizar fuerza bruta sin sanción alguna.
8. ***Software and Data Integrity Failures***. Las fallas en la integridad del *software* y los datos se dan cuando no protegemos nuestro código contra violaciones de integridad durante su creación. Por ejemplo, cuando utilizamos bibliotecas o módulos procedentes de repositorios que no son de confianza.
9. ***Security Logging and Monitoring Failures***. Esta vulnerabilidad consiste en la carencia de monitorización y registro de nuestra aplicación. Si no monitorizamos y registramos los eventos, como podría ser un inicio de sesión, jamás podremos detectar y protegernos ante nuevas amenazas.
10. ***Server-Side Request Forgery (SSRF)***. Esta vulnerabilidad ocurre cuando una aplicación web está obteniendo un recurso externo sin validar la dirección entregada (URL) proporcionada por el usuario.

4. OWASP Juice Shop

OWASP Juice Shop [5] es una aplicación web desarrollada en JavaScript con vulnerabilidades intencionadas para que el usuario pueda explotarlas y aprender sobre seguridad en aplicaciones web (véase Figura 1). El propósito de su creación fue concienciar, y reducir de manera intencionada mediante gamificación, las vulnerabilidades más comunes en seguridad web [6].

4.1. Conexión

Podemos acceder a OWASP Juice Shop a través de diferentes formas: instalando y desplegando en nuestro sistema, levantando un contenedor gracias a herramientas como Docker, etc. En esta ocasión, la desplegaremos en la máquina Kali ejecutando las siguientes órdenes en una terminal:

- `sudo apt install npm`
- `git clone https://github.com/juice-shop/juice-shop.git --depth 1`

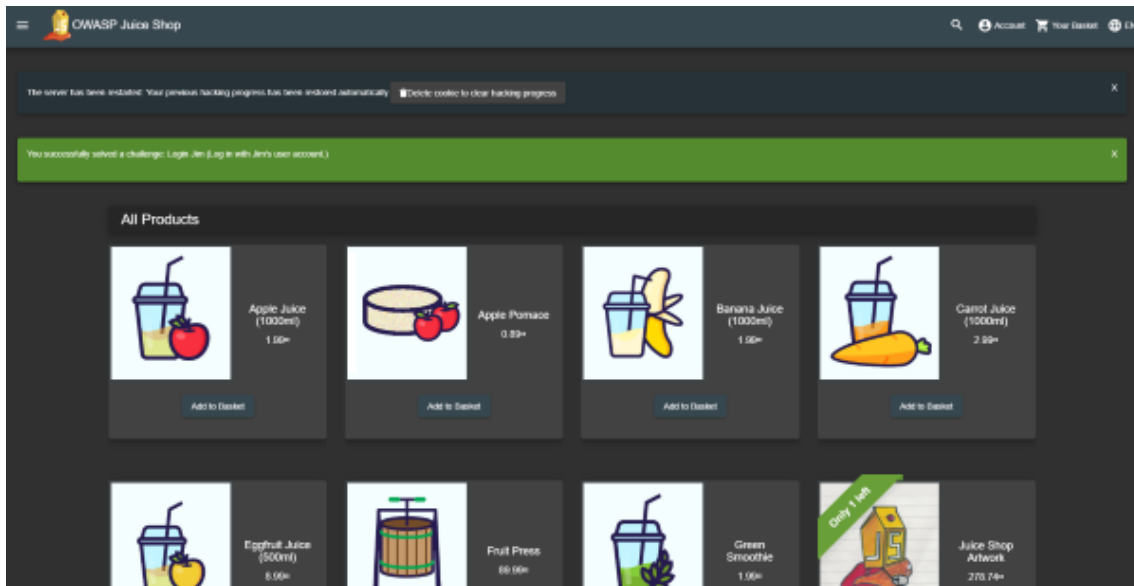


Figura 1: Captura de pantalla de OWASP Juice Shop

- `cd juice-shop`
- `npm install`
- `npm start`
- Ahora juice-shop estará disponible en `http://localhost:3000`

Una vez instalada solo será necesario ejecutar `npm start` en el directorio `juice-shop` cada vez que queramos iniciarla.

5. Burp

Burp [7] es una herramienta para realizar auditorías de seguridad a aplicaciones web desarrollado en Java. Dispone de una versión completa de pago así como otra versión gratuita y reducida en cuanto a funcionalidad. La plataforma cuenta con numerosas herramientas que pone a nuestra disposición:

- *Repeater*. Permite modificar peticiones al servidor así como reenviarlas y observar los resultados.
- *Target*. Permite fijar un objetivo y construir un *SiteMap* a partir de él.
- *Decoder*. Esta herramienta nos permite codificar y decodificar parámetros, URLs, hashes, etc.

- *Spider*. Permite analizar automáticamente las aplicaciones web y mapear sus funciones y contenidos.
- *Intruder*. Permite realizar ataques automáticos haciendo uso de algoritmos que pueden configurarse para generar peticiones HTTP maliciosas.
- Proxy web. Es la funcionalidad más conocida de Burp. Nos brinda de un proxy entre el navegador e Internet que permite interceptar las peticiones e inspeccionar el tráfico.
- *Extender*. Nos permite instalar innumerables extensiones para ampliar las funcionalidades de Burp.

Dentro de Burp las herramientas guardan relación entre sí, como podemos ver en la Figura 2 (imagen extraída de [1]). En esta práctica vamos a trabajar únicamente con la funcionalidad del proxy web.

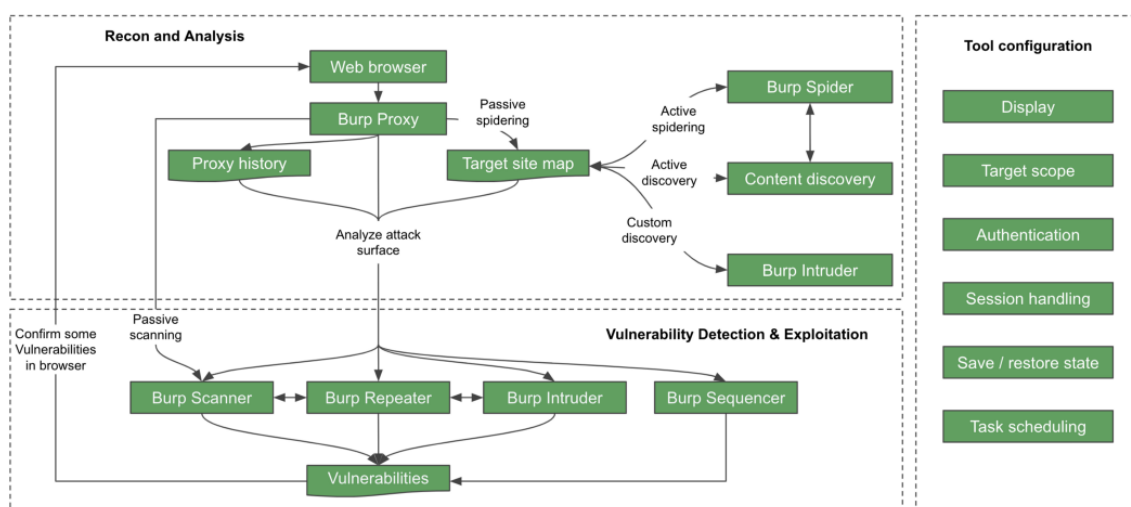


Figura 2: Relación de las herramientas de Burp entre sí

5.1. Proxy web

Un **proxy web** es una herramienta esencial para llevar a cabo el análisis de aplicaciones y servicios web. Una de sus principales funcionalidades es que permite parar las peticiones HTTP/HTTPS realizadas por el navegador con el propósito de analizarlas, manipularlas e incluso enviarlas al servidor. De igual forma, puede utilizarse para parar las respuestas que el servidor envía al navegador.

Por lo tanto, un proxy web nos permite observar y manipular los parámetros de

entrada de una aplicación para llevar a cabo tanto pruebas manuales como automáticas.

5.2. Instalación de Burp

Por defecto, Burp debería estar instalado en nuestra máquina Kali Linux [3]. Para comprobar que se encuentra instalado tan solo debemos abrir el cajón de aplicaciones de Kali Linux y buscar la herramienta Burp (o Burpsuite Community Edition) o en una terminal ingresar el comando:

```
burpsuite --help
```

Y se nos devolvería una salida como la que vemos en la Figura 3.

Si la herramienta no se encuentra instalada en nuestro sistema (el comando no devuelve ninguna salida o no aparece en nuestro cajón de aplicaciones) debemos:

1. Actualizar nuestras dependencias con el comando:

```
sudo apt-get update
```

2. Instalar Burp con el comando:

```
sudo apt install burpsuite
```

3. Si el sistema no lograra localizar el paquete, deberíamos:

- a) Acceder a la web oficial de descarga de Burpsuite Community Edition [8].
- b) Hacer clic en el botón *Download*, teniendo en cuenta que está seleccionada la **Burp Suite Community Edition** y el sistema **Linux (64-bit)**, como vemos en la Figura 4.
- c) Una vez se descargue el archivo, instalamos el paquete gracias al comando:

```
sh nombre_del_paquete.sh
```

Siendo `nombre_del_paquete.sh` el paquete `.sh` que acabamos de descargar. Al ejecutar el comando se abrirá el instalador de Burp como vemos en la Figura 5.

- d) Seguimos las instrucciones del instalador (*Next* → *Next* → *Next*) y esperamos a que termine la instalación. Una vez terminada, podremos acceder a la herramienta desde el cajón de aplicaciones (véase la Sección 5.3).

```
root@kali:~# burpsuite --help
Usage:
--help                Print this message
--version             Print version details
--disable-extensions  Prevent loading of extensions on startup
--diagnostics         Print diagnostic information
--use-defaults        Start with default settings
--collaborator-server Run in Collaborator server mode
--collaborator-config Specify Collaborator server configuration file; defaults to
--data-dir            Specify data directory
--project-file        Open the specified project file; this will be created as a
--config-file         Load the specified project configuration file(s); this option
--user-config-file    Load the specified user configuration file(s); this option
--auto-repair         Automatically repair a corrupted project file specified by
--unpause-spider-and-scanner Do not pause the Spider and Scanner when opening an existing
--disable-auto-update Suppress auto update behavior
```

Figura 3: Salida brindada por el comando burpsuite --help

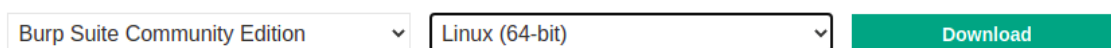


Figura 4: Opciones para descargar Burpsuite Community Edition para Linux 64 bits

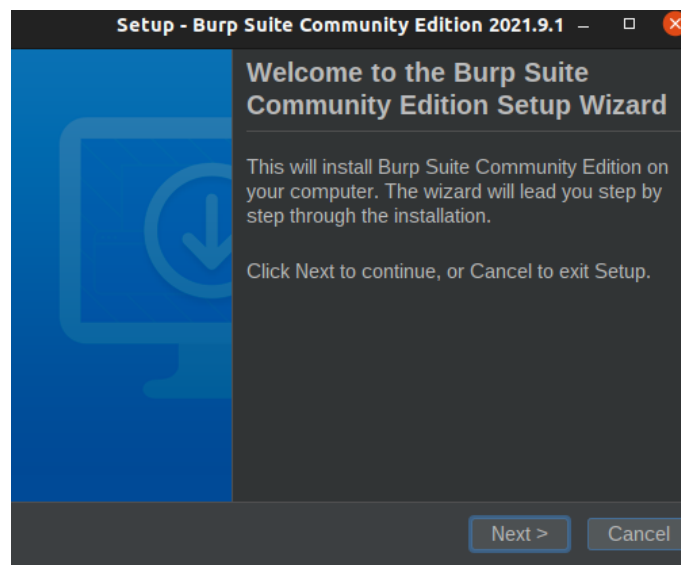


Figura 5: Instalador de la herramienta Burp

5.3. Uso de Burp

Para utilizar Burp como un proxy web, debemos llevar a cabo la siguiente configuración:

1. Ejecutamos *burpsuite*. Si nos apareciese un mensaje sobre términos y condiciones o la versión de Java, pulsamos en aceptar.
2. Creamos un proyecto temporal (véase la Figura 6) y dejamos la opción del fichero de configuración por defecto (véase la Figura 7).
3. Abrimos el navegador que la herramienta pone a nuestra disposición. Para ello, utilizando las pestañas con las diferentes herramientas, navegamos hasta la sección de Proxy. Una vez allí, pulsamos en el botón *Open browser* (véase la Figura 8). Al ser el propio navegador de la aplicación, todo el tráfico ya pasa por el propio proxy de Burp, incluso podemos probarlo sobre HTTPS sin necesidad de instalar certificado alguno. Para comprobarlo, vemos que en la herramienta la opción de interceptación está activada (véase la Figura 9).
4. Para comprobar que el proxy está en funcionamiento realizamos una búsqueda con nuestro navegador web, por ejemplo intentar acceder a `uca.es`, y vemos cómo la herramienta la intercepta (véase la Figura 10).

Para más información sobre el uso de la herramienta de proxy podemos acceder a la documentación oficial [9].

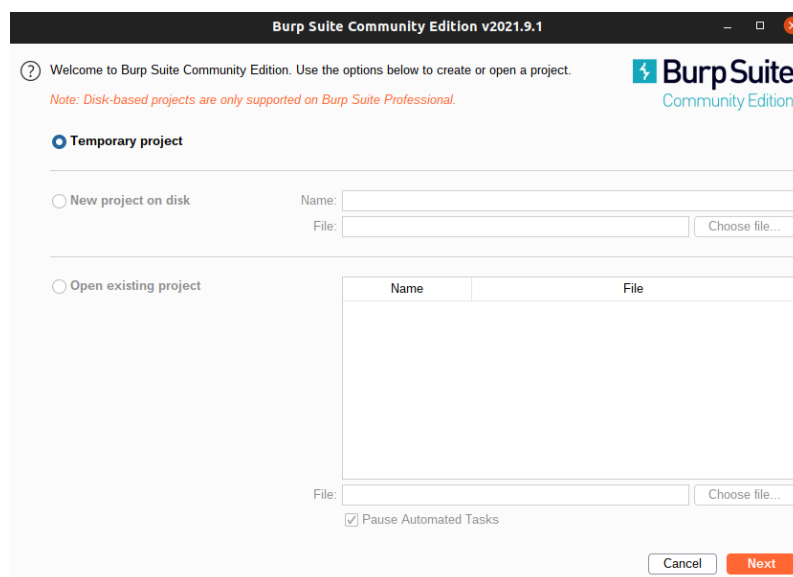


Figura 6: Creamos un proyecto temporal en Burpsuite

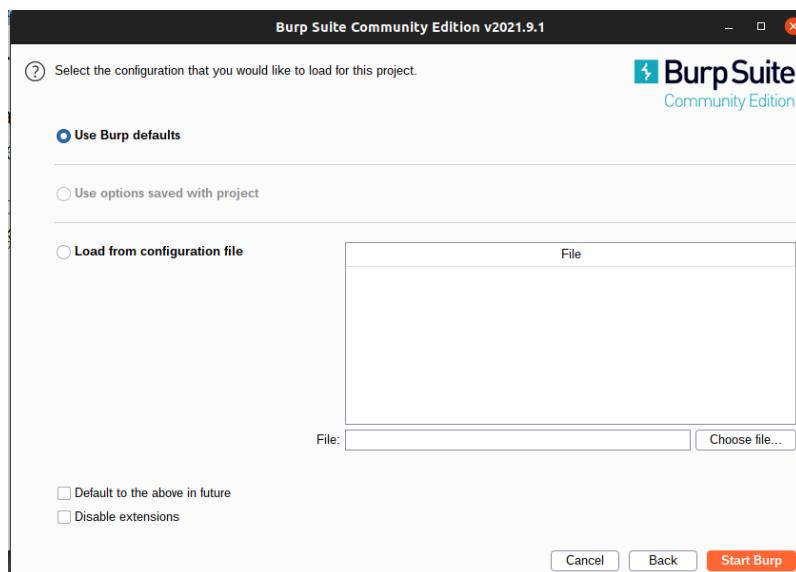


Figura 7: Dejamos la opción por defecto respecto al fichero de configuración

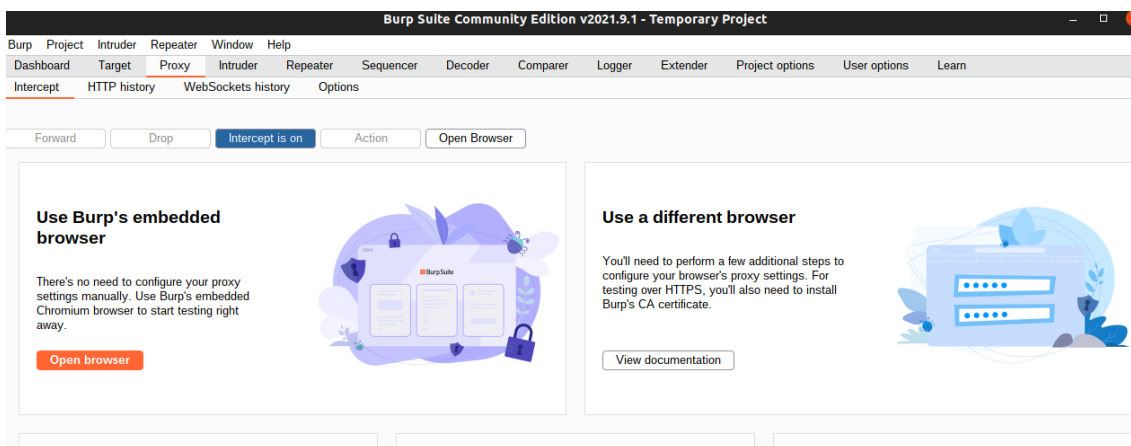


Figura 8: Abrimos el navegador que Burp pone a nuestra disposición

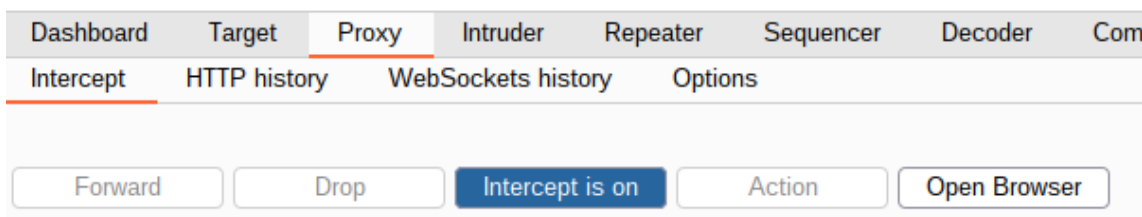


Figura 9: Comprobamos que la opción de *intercept* está en *on*

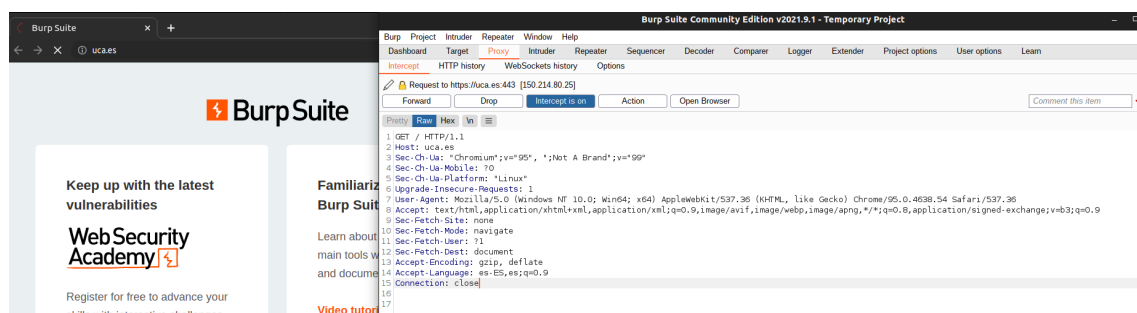


Figura 10: Vemos cómo Burp intercepta nuestra petición

6. Ejercicios

A continuación, realice los siguientes ejercicios.

6.1. Ejercicio 1: OWASP Top 10 - 2021

Atendiendo al OWASP Top 10 - 2021 y realizando búsquedas en Internet, describa un ejemplo real (sin importar hace cuánto haya ocurrido) de cada tipo de ataque.

6.2. Ejercicio 2: Explotación de OWASP Juice Shop

Realizando la conexión vista en la Sección 4.1, realice los siguientes ataques y responda a las preguntas propuestas:

6.2.1. Injection

Vamos a comenzar realizando ataques de inyección hacia la aplicación web. Las vulnerabilidades relacionadas con la inyección son bastante peligrosas para una aplicación, ya que suelen causar tiempo de inactividad y pérdida de datos. Para comenzar con este ataque:

1. Copiamos la dirección IP (véase la Figura 11) de la máquina.
2. Instalamos la extensión del navegador **FoxyProxy** [2].
3. Una vez instalada la extensión debemos configurarla para utilizar Burp (para esto damos clic en la opción *options* que vemos al pulsar en el icono de la extensión en la esquina superior derecha de nuestro navegador, véase la Figura 12).
4. Pulsamos la opción *Add* y rellenamos los campos con los datos que vemos en la Figura 13, que son los que Burpsuite utiliza por defecto.

5. Si pensamos en realizar una inyección, es posible que lo primero que se nos venga a la cabeza sea una inyección de código SQL, y seguramente el primer lugar que se nos ocurra indagar sea en el *login* de la aplicación, por lo que accedemos a la sección de *login* de *OWASP Juice Shop*. Para ello pulsamos en el botón de la esquina superior derecha (*Account*) o hacemos clic en la URL correspondiente: `http://localhost:3000/#/login`
6. Cambiamos el proxy a utilizar dentro de **FoxyProxy** (véase la Figura 14). Y a continuación abrimos la aplicación de Burp asegurándonos que la opción de intercepción está en *intercept is on*.
7. Dentro de la pestaña de *login* que hemos abierto anteriormente introducimos un usuario y contraseña para interceptar la petición (véase la Figura 15) y pulsamos el botón de *login*.
8. Al comprobar de nuevo la herramienta Burp vemos cómo ha interceptado la petición de forma satisfactoria. Dentro de la información que nos refleja podemos ver la dirección y puerto desde donde procede la petición, la forma en la que se transmiten los datos (en este caso texto plano en JSON) y la extensión del mensaje, entre otros datos. Si pulsamos en botón *Forward* (debajo de la dirección de la petición) podremos comprobar cómo aparecen nuevos datos como la información que hemos enviado (quizás sea necesario pulsarlo más de una vez; véase la Figura 16).
9. Ahora que ya sabemos de qué manera se transmite la información, e incluso tenemos un ejemplo con los datos que hemos enviado, podemos utilizar la utilidad *Repeater* de Burp para realizar nuestra inyección de forma exitosa. Lo primero será copiar todos los datos de la petición que acabamos de visualizar.
10. Ahora navegamos hacia la herramienta *Repeater* (dos hacia la derecha de la herramienta Proxy), y pegamos los datos de la petición. Para realizar el ataque podemos cambiar el usuario por una tautología (véase la Figura 17). Una vez hecho esto, devolvemos la petición haciendo clic en el botón *Send* (si se nos requiriese la dirección de la petición podemos comprobarla desde la herramienta *Proxy*), y vemos como se nos devuelve el *email* del administrador (véase la Figura 18).
11. Esta respuesta nos hace saber que, gracias a nuestra tautología podremos iniciar sesión. Regresamos a la pestaña de *login* e ingresamos la tautología como usuario (para mayor comodidad ya podemos cambiar Burp hacia *intercept is off*). Hemos conseguido acceso a la cuenta como vemos en la Figura 19.

Con este ejemplo de ataque como base, responda a las siguientes preguntas:

Active Machine Information			
Title OWASP-Juice-Shop	IP Address 10.10.157.180	Expires 57m 10s	? Add 1 hour Terminate

Figura 11: Vemos la IP de la máquina

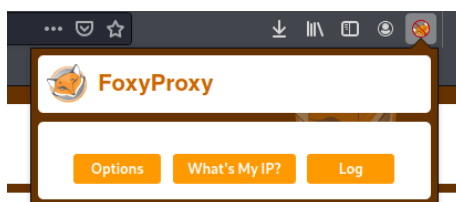


Figura 12: Hacemos clic en el icono de FoxyProxy

Title or Description (optional)
Burp

Color
#cc7373

Pattern Shortcuts
Enabled
Add whitelist pattern to match all URLs
Do not use for localhost and intranet/private IP addresses

Proxy Type
HTTP

Proxy IP address or DNS name
127.0.0.1

Port
8080

Username (optional)
username

Password (optional)

On

On

Off

Cancel

Save & Add Another

Save & Edit Patterns

Save

Figura 13: Añadimos los datos para la conexión del Proxy de Burp

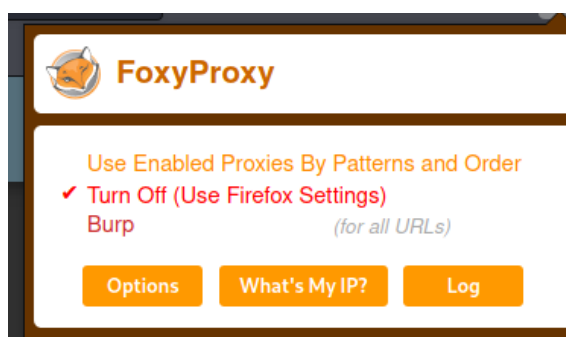


Figura 14: Cambiamos nuestro proxy para utilizar Burp

- ¿Por qué la base de datos nos devuelve ese resultado cuando le enviamos una tautología?

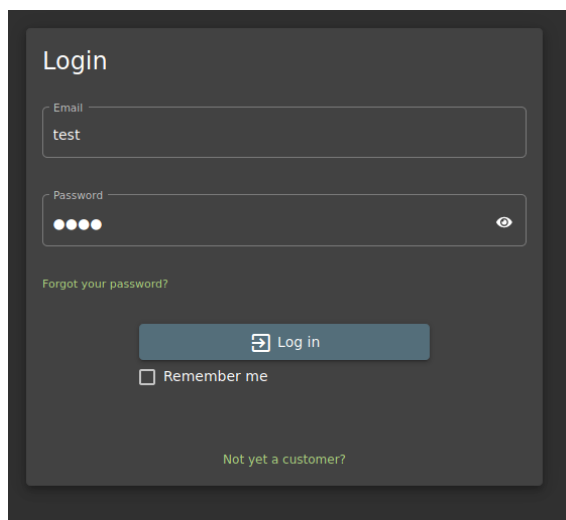


Figura 15: Introducimos datos en la pestaña de *login*

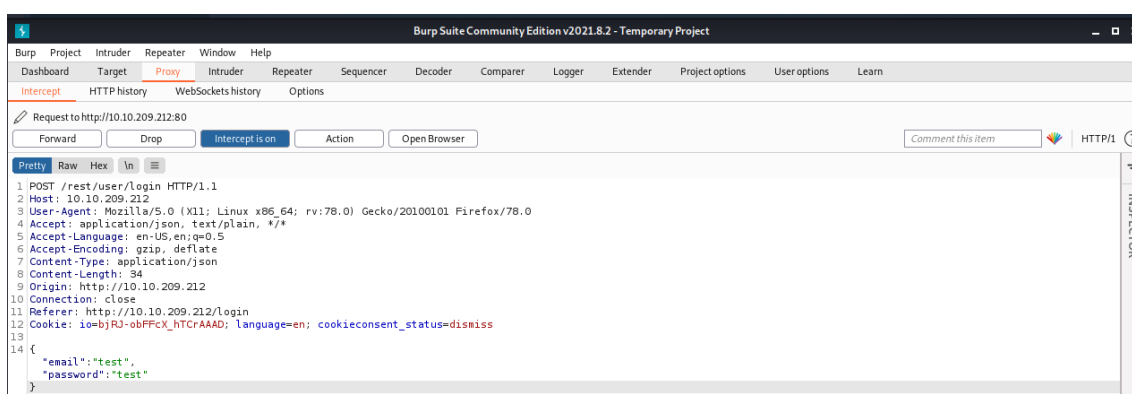


Figura 16: Aparecen los datos de nuestra petición al pulsar el botón *forward*

- ¿Por qué nunca deberíamos enviar en nuestra aplicación los datos en texto plano?
- Siguiendo el procedimiento anterior, ¿cómo podríamos conseguir acceso a la cuenta de `bender@juice-sh.op`?

6.2.2. Broken Authentication

De nuevo, utilizando Burp vamos a realizar un ataque hacia la autenticación de la web. Para ello:

1. Copiamos la dirección IP (véase la Figura 11) de la máquina. Si juice-shop

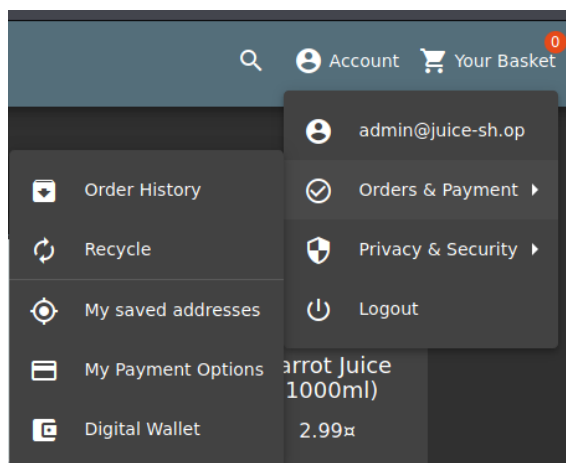


Figura 19: Hemos conseguido acceso a la cuenta del administrador

está funcionando en nuestra máquina, será *localhost* o 127.0.0.1.

2. Instalamos la extensión del navegador **FoxyProxy** [2].
3. Una vez instalada la extensión debemos configurarla para utilizar Burp (para esto damos clic en la opción *options* que vemos al pulsar en el icono de la extensión en la esquina superior derecha de nuestro navegador, véase la Figura 12).
4. Pulsamos la opción *Add* y rellenamos los campos con los datos que vemos en la Figura 13, que son los que Burpsuite utiliza por defecto.
5. Vamos hacia la sección de *login* y abrimos la aplicación Burp.
6. Instalamos las listas de contraseñas a probar gracias al comando:

```
sudo apt-get install seclists
```

Podemos encontrar estas listas, y concretamente la que vamos a utilizar para este ataque en:

```
/usr/share/seclists/Passwords/Common-Credentials/best1050.txt
```

7. Dentro de la aplicación Burp navegamos hacia la sección *Intruder* y localizamos la opción de *Payload Options [Simple list]* (véase la Figura 20). Hacemos clic en el botón de *Load* y navegamos hasta la ruta nombrada en el punto anterior para seleccionar la lista *best11050.txt* (véase la Figura 21).
8. Una vez cargado el *payload*, dentro de *Intruder* vamos hacia la sección de

Target e introducimos los datos del *host* y el puerto (véase la Figura 22).

9. Dentro de la sección *Positions* pegamos los datos que la sección de *Proxy* nos brinda y sustituimos el *password* por `$test$` haciendo uso del botón *Add* que vemos a la izquierda de la pantalla. El resultado final será parecido a lo que vemos en la Figura 23.
10. Hacemos clic en el botón *Start attack*. Dejamos que la aplicación trabaje hasta encontrar un *password* correcto. Como podemos ver, la aplicación nos arroja un código de *Status* del que podemos saber más haciendo clic en cada contraseña (véase la Figura 24):
 - 401. *Unauthorized*.
 - 200. *Authorized*
11. Hemos encontrado que `admin123` es la contraseña correcta. Si probamos a hacer *login* veremos que, efectivamente, podemos acceder.

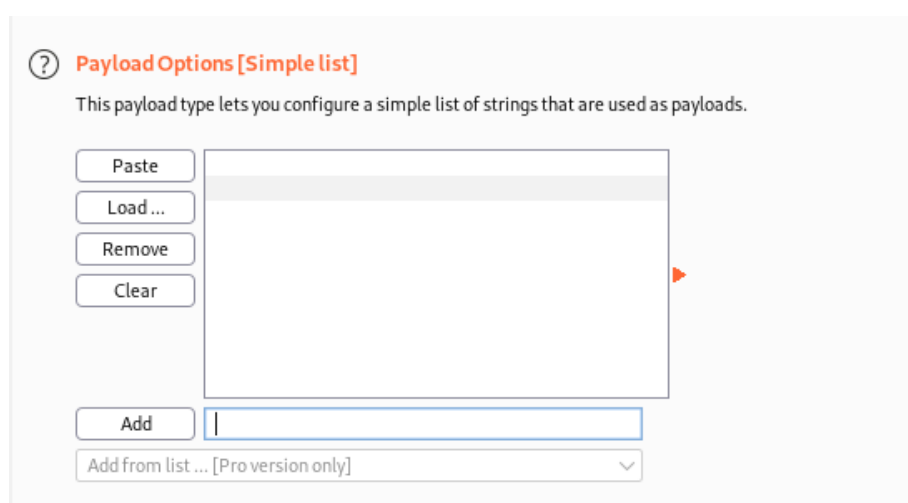


Figura 20: Localizamos la sección de Payload Options

Una vez realizado este ataque, responde a las siguientes preguntas:

- ¿Podríamos haber realizado este ataque si en nuestra aplicación hubiera algún tipo de tecnología para evitar la fuerza bruta?
- Dentro de la vulnerabilidad conocida como *Broken Authentication* también podemos explotar la sección de toda página web: ¿Has olvidado tu contraseña? Para demostrarlo vamos a cambiar la contraseña de `jim@juice-sh.op` sin saber su contraseña actual (pista: quizás la búsqueda en Google de *Jim Star Trek* pueda ayudarte):

Práctica 6: Explotación de aplicaciones web y bases de datos

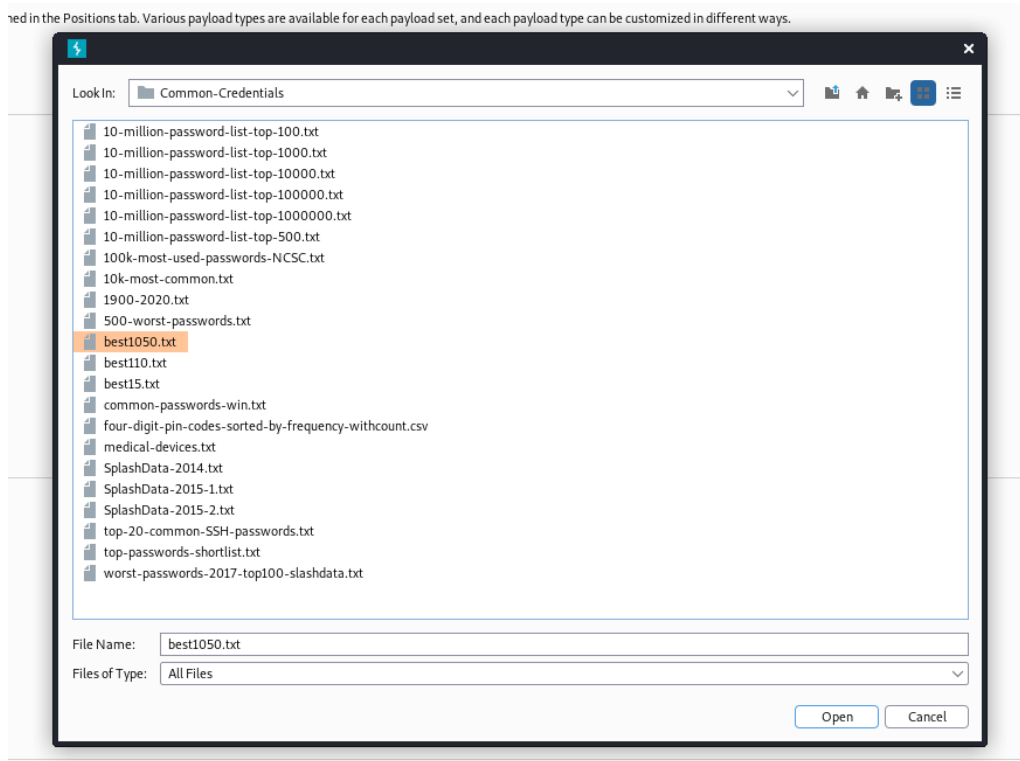


Figura 21: Seleccionamos la lista best1050

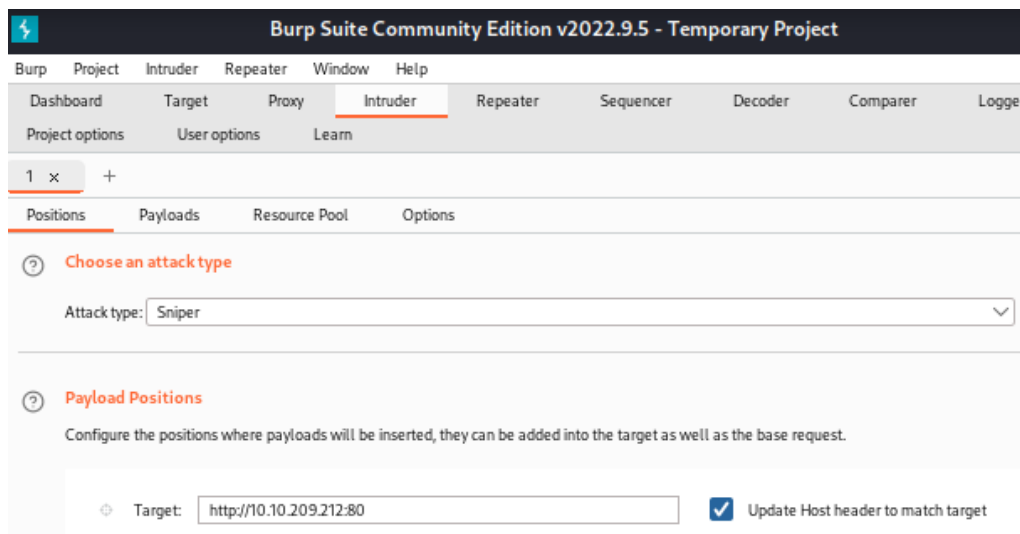
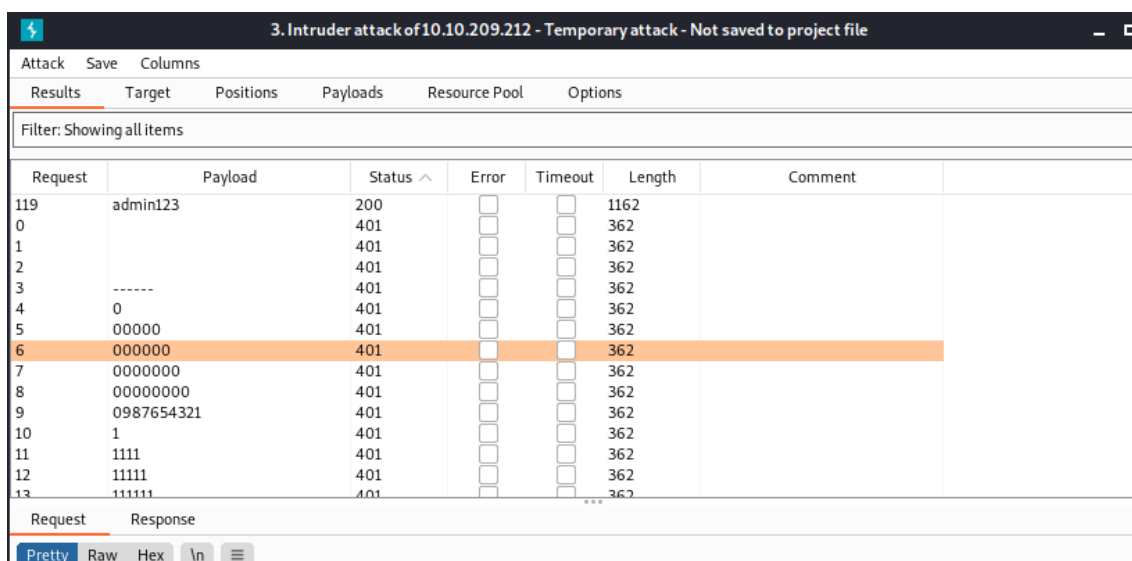


Figura 22: Colocamos los datos del target

Práctica 6: Explotación de aplicaciones web y bases de datos

```
1 POST /rest/user/login HTTP/1.1
2 Host: 10.10.209.212
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0
4 Accept: application/json, text/plain, */*
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Content-Type: application/json
8 Content-Length: 47
9 Origin: http://10.10.209.212
10 Connection: close
11 Referer: http://10.10.209.212/login
12 Cookie: io=bjRJ-obFFcX_hTCrAAAD; language=en; cookieconsent_status=dismiss
13
14 {"email":"admin@juice-sh.op","password":"$test$"}
```

Figura 23: Modificamos el *password* para utilizar la lista de contraseñas



Request	Payload	Status	Error	Timeout	Length	Comment
119	admin123	200	<input type="checkbox"/>	<input type="checkbox"/>	1162	
0		401	<input type="checkbox"/>	<input type="checkbox"/>	362	
1		401	<input type="checkbox"/>	<input type="checkbox"/>	362	
2		401	<input type="checkbox"/>	<input type="checkbox"/>	362	
3	-----	401	<input type="checkbox"/>	<input type="checkbox"/>	362	
4	0	401	<input type="checkbox"/>	<input type="checkbox"/>	362	
5	00000	401	<input type="checkbox"/>	<input type="checkbox"/>	362	
6	000000	401	<input type="checkbox"/>	<input type="checkbox"/>	362	
7	0000000	401	<input type="checkbox"/>	<input type="checkbox"/>	362	
8	00000000	401	<input type="checkbox"/>	<input type="checkbox"/>	362	
9	0987654321	401	<input type="checkbox"/>	<input type="checkbox"/>	362	
10	1	401	<input type="checkbox"/>	<input type="checkbox"/>	362	
11	1111	401	<input type="checkbox"/>	<input type="checkbox"/>	362	
12	11111	401	<input type="checkbox"/>	<input type="checkbox"/>	362	
13	111111	401	<input type="checkbox"/>	<input type="checkbox"/>	362	

Figura 24: Coincidencia encontrada en la línea 117 de nuestro archivo (admin123)

1. ¿Cuál es la pregunta de seguridad de Jim?
2. ¿Cómo se llamaba el hermano de Jim?
3. ¿Qué podríamos hacer para prevenir que otros usuarios que no sean el propietario del correo sepan la pregunta de seguridad de una cuenta de nuestra aplicación?
4. Adjunta una captura mostrando que, efectivamente has conseguido el acceso a la cuenta de Jim.

6.2.3. Sensitive Data Exposure

Dentro de una aplicación web se guarda y se transmite información sensible que debería ser almacenada de la forma correcta. No obstante, en algunas ocasiones los desarrolladores podrían no almacenarla donde deberían e incluso hacerla visible para todos, consiguiendo que esta información sea vulnerable.

Podemos ver un ejemplo claro si accedemos a la sección de *About me* de *OWASP Juice Shop*: <http://localhost:3000/login#/about>

A primera vista parece que no ocurre nada en esta sección; sin embargo, si pasamos el ratón por el enlace (color verde) de las palabras *Check out our boring terms of use if you are interested in such lame stuff* podemos ver algo que llama nuestra atención (véase la Figura 25). Con esta información, responda a las siguientes preguntas:

- ¿Cuál es la dirección del enlace?
- ¿Qué pasa si hacemos clic en el enlace?
- Teniendo en cuenta la dirección del enlace, ¿cómo podríamos aprovechar esta vulnerabilidad?
- Una vez accedido al servidor FTP, descargue el archivo `acquisitions.md` y copie la última línea para dar respuesta a esta pregunta.



Figura 25: Comprobamos que el enlace nos llevaría hacia una dirección FTP

Referencias

- [1] ArtsSEC: *Introducción a Burp Suite*. <https://medium.com/@ArtsSEC/introduccion-a-burp-suite-4f3d14b32af>, visitado el 21-11-2024.
- [2] Eric H. Jung: *Extensión FoxyProxy para Firefox*. <https://addons.mozilla.org/es/firefox/addon/foxyproxy-standard/>, visitado el 21-11-2024.
- [3] OffSec Services Limited: *Burpsuite Kali Linux Suite*. <https://www.kali.org/tools/burpsuite/>, visitado el 21-11-2024.
- [4] OWASP: *Open Web Application Security Project*. https://www.owasp.org/index.php/Main_Page, visitado el 21-11-2024.
- [5] OWASP: *OWASP Juice Shop en GitHub*. <https://github.com/OWASP/www-project-juice-shop/blob/master/index.md>, visitado el 21-11-2024.
- [6] OWASP: *OWASP Top Ten 2021*. <https://owasp.org/www-project-top-ten/>, visitado el 21-11-2024.
- [7] Port Swigger: *Burp web oficial*. <https://portswigger.net/burp>, visitado el 21-11-2024.
- [8] Port Swigger: *Burpsuite Community Edition Download*. <https://portswigger.net/burp/releases/professional-community-2021-9-1?requestededition=community>, visitado el 21-11-2024.
- [9] Port Swigger: *Getting started with Burp Proxy*. <https://portswigger.net/burp/documentation/desktop/tools/proxy/getting-started>, visitado el 21-11-2024.