

Informe Laboratorio 2

Sección 03

Anselmo Pacheco

e-mail: anselmo.pacheco@mail.udp.cl

13 de septiembre de 2024

Índice

1. Descripción de actividades	2
2. Desarrollo de actividades según criterio de rúbrica	3
2.1. Levantamiento de docker para correr DVWA (dvwa)	3
2.2. Redirección de puertos en docker (dvwa)	4
2.3. Obtención de consulta a replicar (burp)	6
2.4. Identificación de campos a modificar (burp)	7
2.5. Obtención de diccionarios para el ataque (burp)	7
2.6. Obtención de al menos 2 pares (burp)	11
2.7. Obtención de código de inspect element (curl)	13
2.8. Utilización de curl por terminal (curl)	14
2.9. Demuestra 4 diferencias (curl)	15
2.10. Instalación y versión a utilizar (hydra)	19
2.11. Explicación de comando a utilizar (hydra)	19
2.12. Obtención de al menos 2 pares (hydra)	20
2.13. Explicación paquete curl (tráfico)	21
2.14. Explicación paquete burp (tráfico)	22
2.15. Explicación paquete hydra (tráfico)	22
2.16. Mención de las diferencias (tráfico)	23
2.17. Detección de SW (tráfico)	24
2.18. Interacción con el formulario (python)	24
2.19. Cabeceras HTTP (python)	24
2.20. Obtención de al menos 2 pares (python)	24
2.21. Comparación de rendimiento con Hydra, Burpsuite, y cURL (python)	24
2.22. Demuestra 4 métodos de mitigación (investigación)	25

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?
- Desarrolle un script en Python para realizar un ataque de fuerza bruta:
 - Utilice la librería requests para interactuar con el formulario ubicado en vulnerabilities/brute y desarrollar su propio script de fuerza bruta en Python. El script debe realizar intentos de inicio de sesión probando una lista de combinaciones de usuario/contraseña.
 - Identifique y explique la cabecera HTTP que empleará para realizar el ataque de fuerza bruta.
 - Muestre el código y los resultados obtenidos (al menos 2 combinaciones válidas de usuario/contraseña).
 - Compare el rendimiento de este script en Python con las herramientas Hydra, Burpsuite, y cURL en términos de velocidad y detección.
- Investigue y describa 4 métodos comunes para prevenir o mitigar ataques de fuerza bruta en aplicaciones web:
 - Para cada método, explique su funcionamiento, destacando en qué escenarios es más eficaz.

2. Desarrollo de actividades según criterio de rúbrica

2.1. Levantamiento de docker para correr DVWA (dvwa)

El proceso realizado para el levantamiento de docker para ejecutar DVWA en el navegador corresponde al siguiente comando que permite la ejecución de docker:

```
sudo docker run --rm -it -p 80:80 vulnerables/web-dvwa
```

```
anselmo@hp-pavilion:~$ sudo docker run --rm -it -p 80:80 vulnerables/web-dvwa
[sudo] contraseña para anselmo:
[+] 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 <==
[Sun Sep 17 18:40:44.844435 2023] [mpm_prefork:notice] [pid 313] AH00163: Apache
/2.4.25 (Debian) configured -- resuming normal operations
[Sun Sep 17 18:40:44.844592 2023] [core:notice] [pid 313] AH00094: Command line:
'/usr/sbin/apache2'

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

==> /var/log/apache2/access.log <==
172.17.0.1 - - [17/Sep/2023:18:41:09 +0000] "GET /vulnerabilities/brute/?username=snm&password=kwn&Login=Login HTTP/1.1" 302 343 "http://172.17.0.2/vulnerabilities/brute/" "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0 Safari/537.36 OPR/102.0.0.0"
172.17.0.1 - - [17/Sep/2023:18:41:09 +0000] "GET /login.php HTTP/1.1" 200 1049 "http://172.17.0.2/vulnerabilities/brute/" "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0 Safari/537.36 OPR/102.0.0.0"
```

Figura 1: Levantamiento docker.

2 DESARROLLO DE ACTIVIDADES SEGÚN CRITERIO DE RÚBRICA

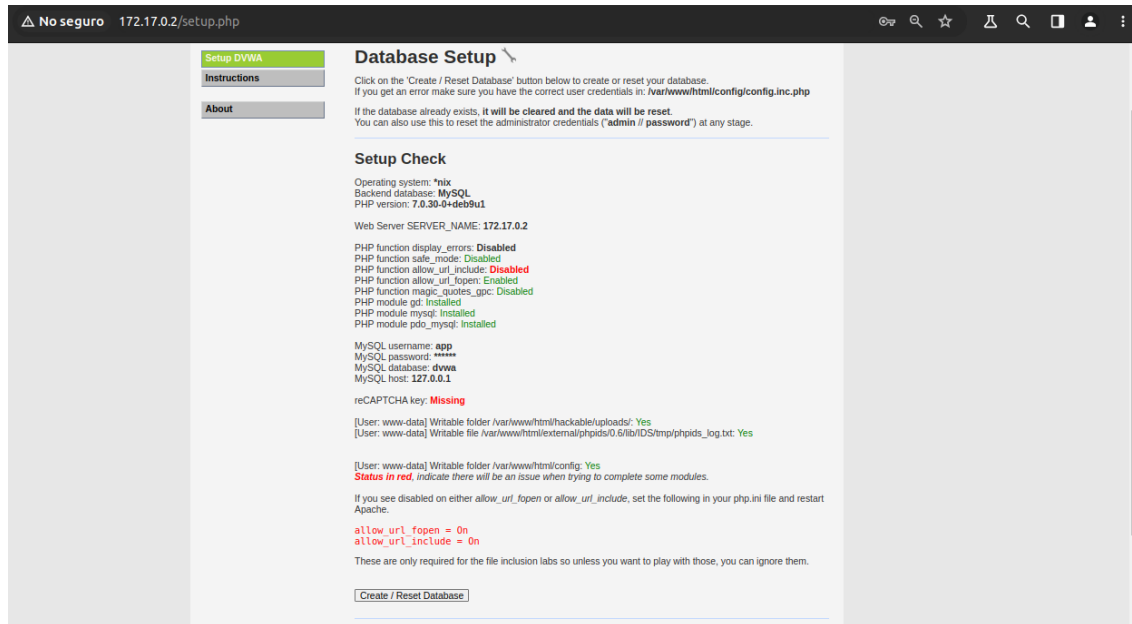


Figura 2: Ejecución comando.

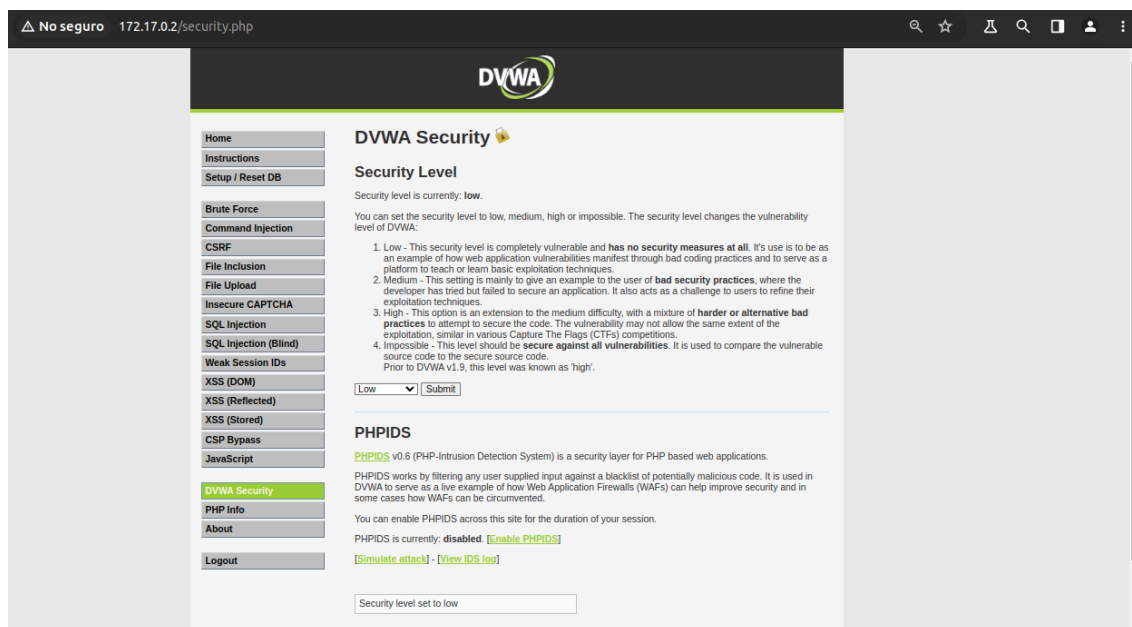


Figura 3: Configuración DVWA.

2.2. Redirección de puertos en docker (dvwa)

Lo que está realizando el comando `sudo docker run -rm -it -p 80:80 vulnerables/web-dvwa` es que se está creando un contenedor Docker utilizando la imagen `vulnerables/web-dvwa`.

dvwa y mapeando el puerto 80 del contenedor al puerto 80 del host. Lo que permite que el tráfico dirigido al puerto 80 sea dirigido al puerto 80 del contenedor docker. Y de esta manera poder acceder a DVWA (Damn Vulnerable Web Application) desde el navegador web.

Para detallar de mejor manera lo que realiza el comando se procederá a explicar paso a paso lo que realiza cada sentencia:

sudo: Este comando permite ejecutar el comando que sigue con privilegios de administrador. Puede ser necesario para ejecutar comandos Docker, dependiendo de la configuración.

docker: Es el comando principal para interactuar con el sistema de contenedores Docker.

run: Indica que queremos ejecutar un nuevo contenedor a partir de una imagen.

-rm: Esta opción indica a Docker que elimine el contenedor automáticamente después de que se detenga. Esto es útil para evitar la acumulación de contenedores inactivos.

-it: Combina dos opciones:

-i: Permite la interactividad del contenedor, lo que significa que puedes interactuar con él desde tu terminal.

-t: Asigna un pseudo-terminal (TTY) al contenedor. Esto es necesario para la interacción interactiva.

-p 80:80: Esta opción establece un mapeo de puertos entre el sistema host y el contenedor. En este caso, el puerto 80 del host se mapea al puerto 80 del contenedor. Esto significa que cualquier tráfico dirigido al puerto 80 del host se redirige al puerto 80 del contenedor.

vulnerables/web-dvwa: Es el nombre de la imagen del contenedor que se utilizará para crear y ejecutar el contenedor. En este caso, se está utilizando una imagen llamada vulnerables/web-dvwa, que es Damn Vulnerable Web Application (DVWA), una aplicación web intencionadamente insegura para propósitos educativos y de pruebas de seguridad.

2 DESARROLLO DE ACTIVIDADES SEGÚN CRITERIO DE RÚBRICA

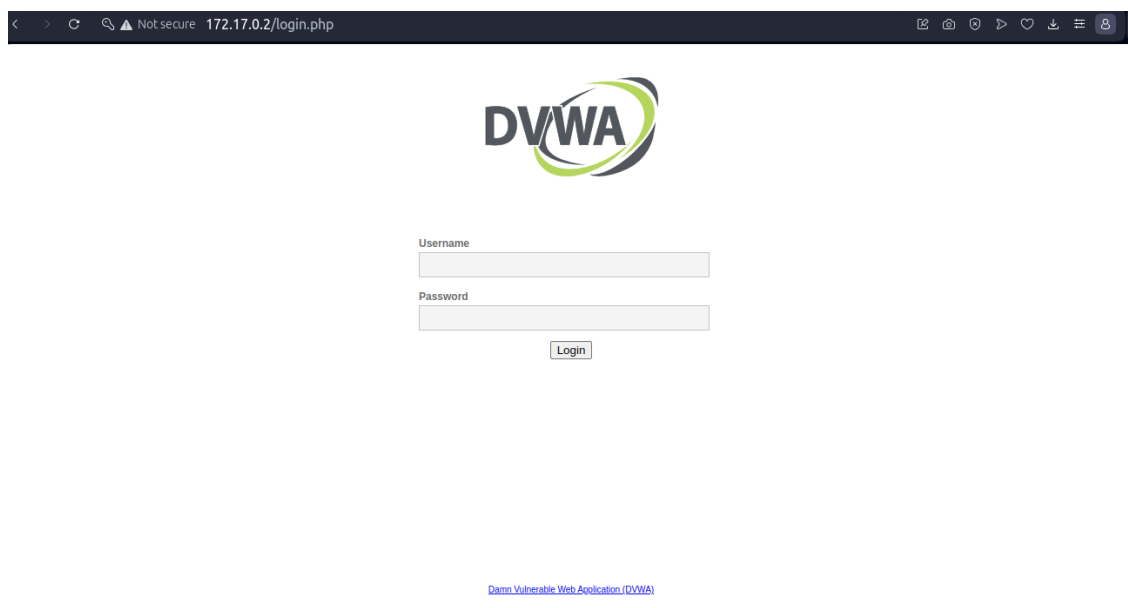


Figura 4: DVWA desde navegador web.

2.3. Obtención de consulta a replicar (burp)

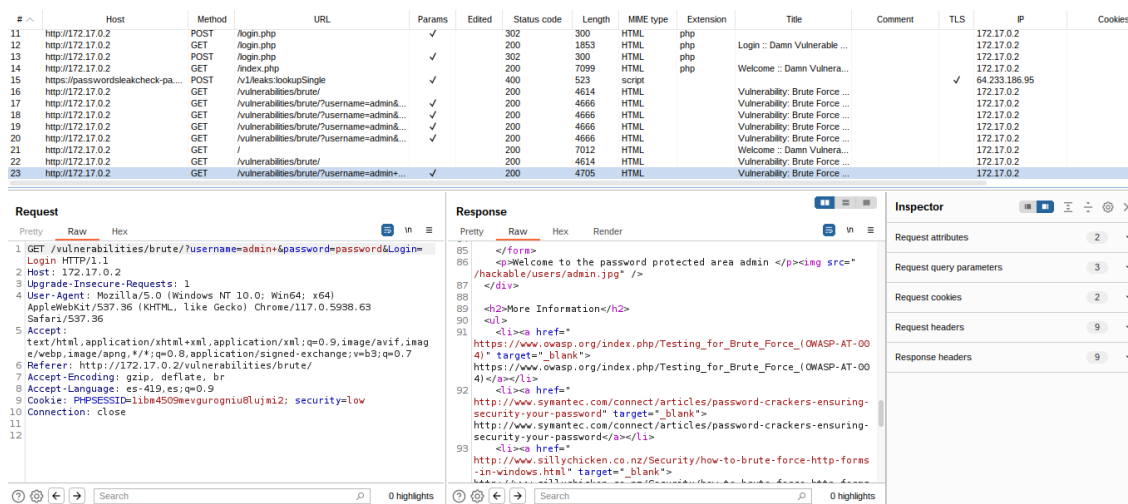


Figura 5: Consulta a replicar

Lo cual permitirá identificar los campos y secciones que se deben observar para el desarrollo del ataque por fuerza bruta.

2.4. Identificación de campos a modificar (burp)

En este punto se proceden a identificar los campos que deben marcarse para luego, al momento de realizar el intruder se permitan reemplazar estos campos seleccionados por el valor del diccionario creado para cada campo.

Los campos seleccionados son: admin y password.

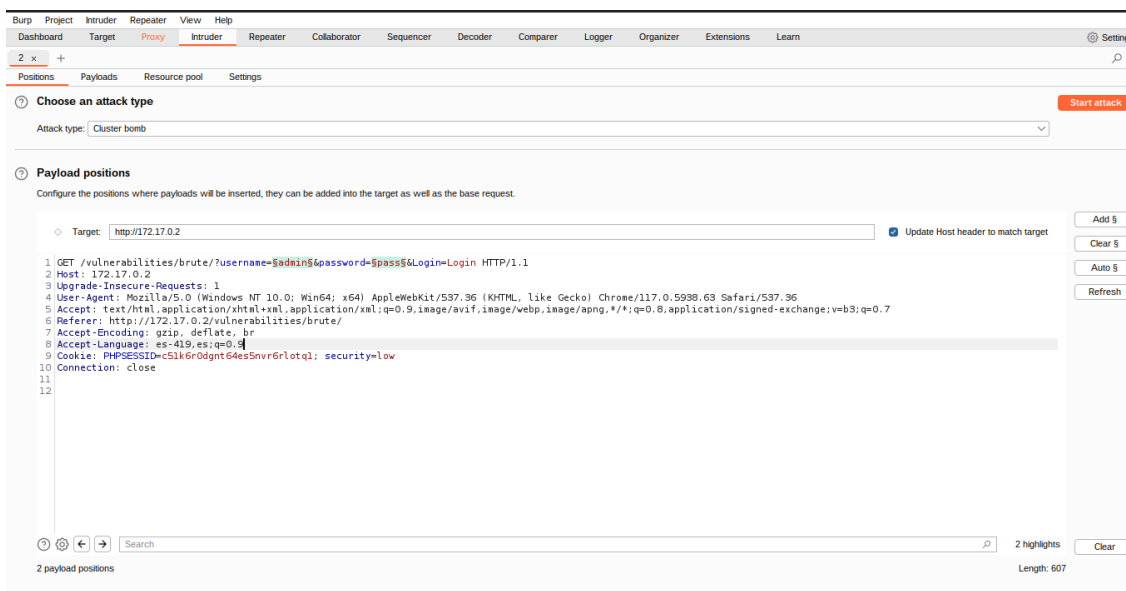


Figura 6: Selección de campos

2.5. Obtención de diccionarios para el ataque (burp)

En este punto se proceden a crear dos archivos que contengan posibles nombres de usuario y contraseña, en donde de esta manera se podrán crear diccionarios que serán utilizados para realizar el ataque de fuerza bruta.

Para realizar este ataque se procederán a reemplazar cada valor del diccionario Usuarios.txt para cada diccionario Contraseñas.txt

Los diccionarios creados son los siguientes:

```

GNU nano 6.2                               Usuario
admin
user
guest
test
demo
root
manager
support
teacher
student
smithy
cr7
messi
supervisor
finance_user
it_admin
project_manager
marketing_user
operations_user
engineering_user
[ 20 líneas escritas ]
^G Ayuda      ^O Guardar    ^W Buscar     ^K Cortar     ^T Ejecutar   ^C Ubicación
^X Salir      ^R Leer fich. ^_ Reemplazar  ^U Pegar      ^J Justificar ^_ Ir a línea
  
```

```

GNU nano 6.2                               Usuario
admin
user
guest
test
demo
root
manager
support
teacher
student
smithy
cr7
messi
supervisor
finance_user
it_admin
project_manager
marketing_user
operations_user
engineering_user
[ 20 líneas escritas ]
^G Ayuda      ^O Guardar    ^W Buscar     ^K Cortar     ^T Ejecutar   ^C Ubicación
^X Salir      ^R Leer fich. ^_ Reemplazar  ^U Pegar      ^J Justificar ^_ Ir a línea
  
```

Figura 7: Diccionario Contraseñas

Luego de la creación de usuarios y contraseñas posibles se procederán a cargar los archivos en burp suite para realizar un ataque de fuerza bruta, el cuál será en este caso el ataque “Cluster Bomb” que corresponde a una técnica de prueba de fuerza bruta que combina múltiples listas de palabras para generar una lista de contraseñas potenciales. Funciona tomando una lista de palabras y combinándola con otra lista de palabras, generando así todas las combinaciones posibles entre los elementos de ambas listas. Para cada payload se cargarán los datos de los diccionarios para ser reemplazados en los campos seleccionados anteriormente, quedando de la siguiente manera:

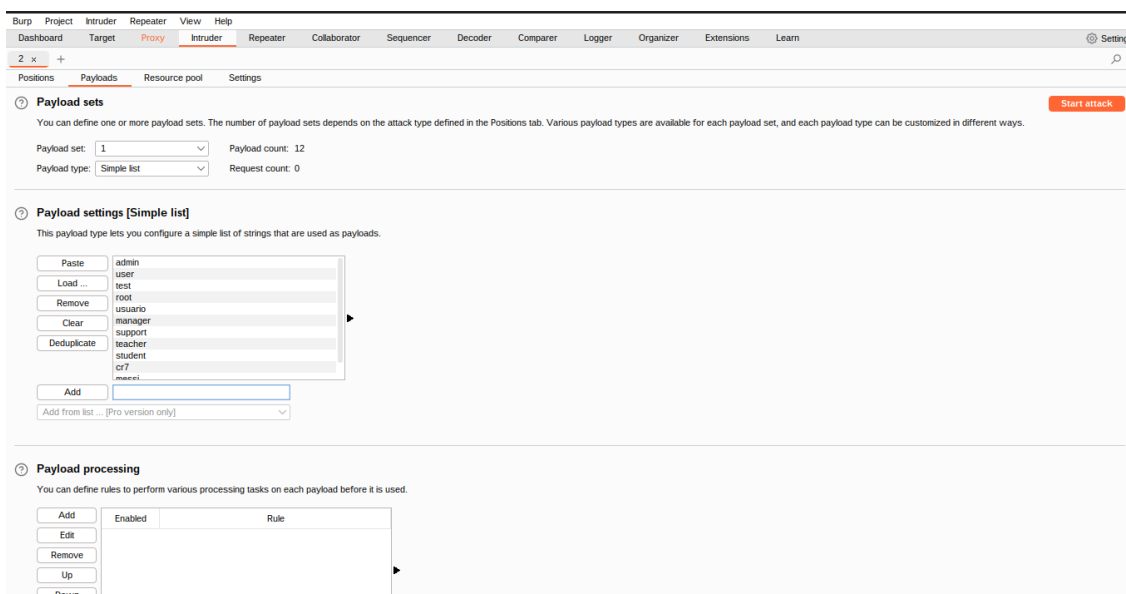


Figura 8: Reemplazo payload usuarios

2 DESARROLLO DE ACTIVIDADES SEGÚN CRITERIO DE RÚBRICA

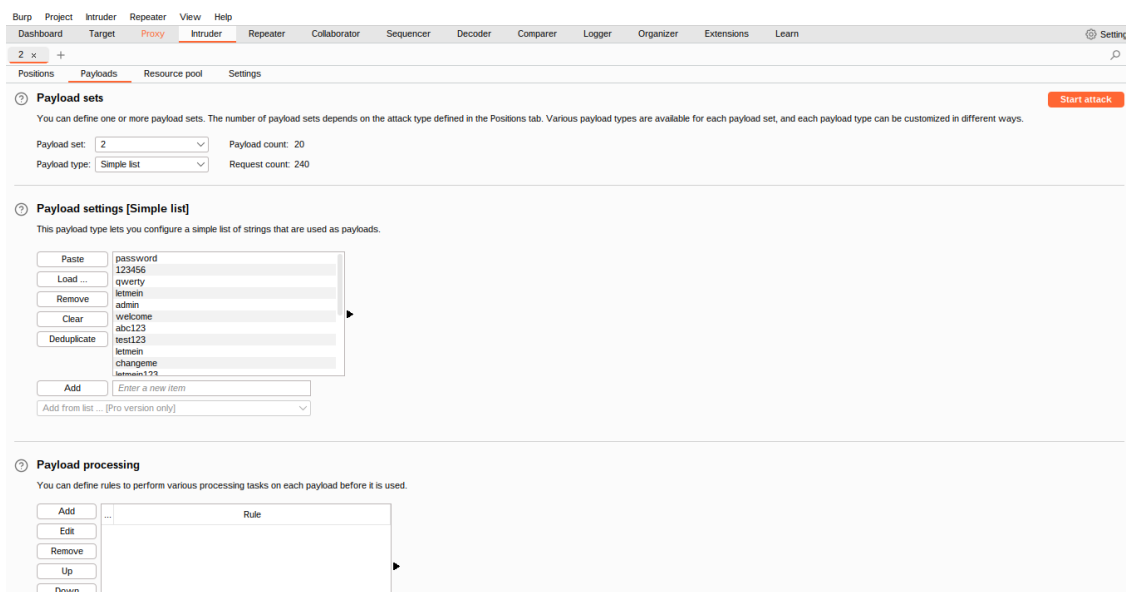


Figura 9: reemplazo payload contraseñas

Una vez realizado el ingreso de los archivos (diccionarios) creados se debe realizar el ataque por fuerza bruta.

The screenshot shows the Burp Suite Results window. The top menu bar includes Attack, Save, and Columns. Below the menu, there are tabs for Results (selected), Positions, Payloads, Resource pool, and Settings. The main window displays a table of results with the following columns: Request, Payload 1, Payload 2, Status code, Error, Timeout, Length, and Comment. The table shows 11 requests, each with a status code of 200 and a length of 4704. The payloads are combinations of usernames and passwords. A red progress bar at the bottom indicates the attack is finished.

Request	Payload 1	Payload 2	Status code	Error	Timeout	Length	Comment
0			200	<input type="checkbox"/>	<input type="checkbox"/>	4704	
1	admin	password	200	<input type="checkbox"/>	<input type="checkbox"/>	4742	
2	user	password	200	<input type="checkbox"/>	<input type="checkbox"/>	4704	
3	test	password	200	<input type="checkbox"/>	<input type="checkbox"/>	4704	
4	root	password	200	<input type="checkbox"/>	<input type="checkbox"/>	4703	
5	usuario	password	200	<input type="checkbox"/>	<input type="checkbox"/>	4704	
6	manager	password	200	<input type="checkbox"/>	<input type="checkbox"/>	4704	
7	support	password	200	<input type="checkbox"/>	<input type="checkbox"/>	4704	
8	teacher	password	200	<input type="checkbox"/>	<input type="checkbox"/>	4703	
9	student	password	200	<input type="checkbox"/>	<input type="checkbox"/>	4704	
10	cr7	password	200	<input type="checkbox"/>	<input type="checkbox"/>	4704	
11	messi	password	200	<input type="checkbox"/>	<input type="checkbox"/>	4704	

Figura 10: Ataque Cluster Bomb

2.6. Obtención de al menos 2 pares (burp)

Una vez finalizado el ataque se deben a comenzar a revisar parámetros que permitan determinar si el usuario y contraseña realizado en la combinación permite ingresar de manera correcta.

Para este caso se procede a realizar un filtro con la palabra "Welcome" que permite identificar que combinación es bienvenida en el sitio web, en donde se obtuvo lo siguiente:

Se proceden a adjuntar los pares obtenidos.

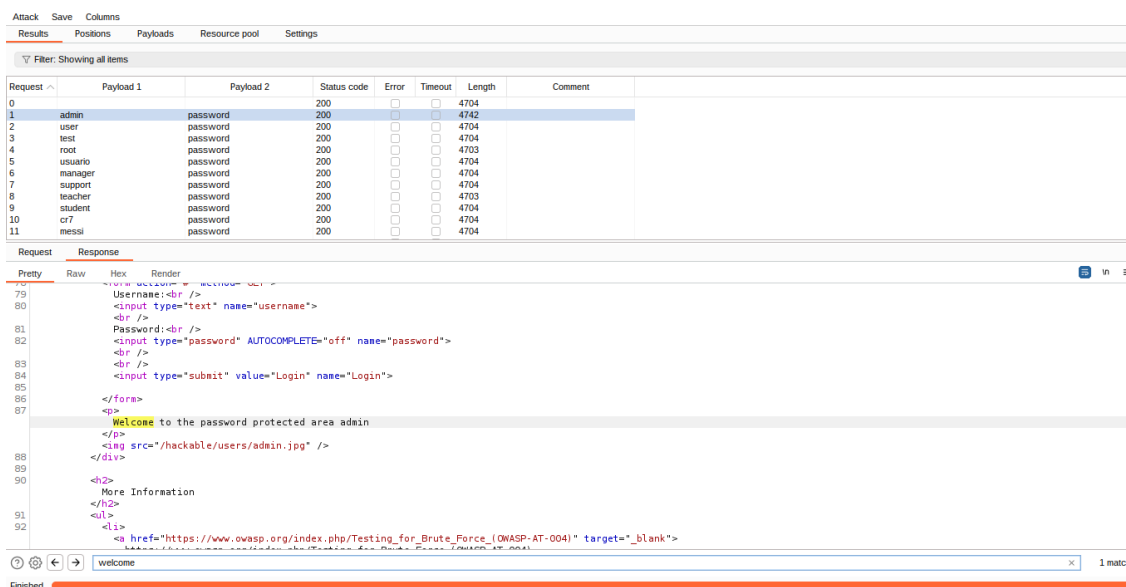


Figura 11: Par válido número 1

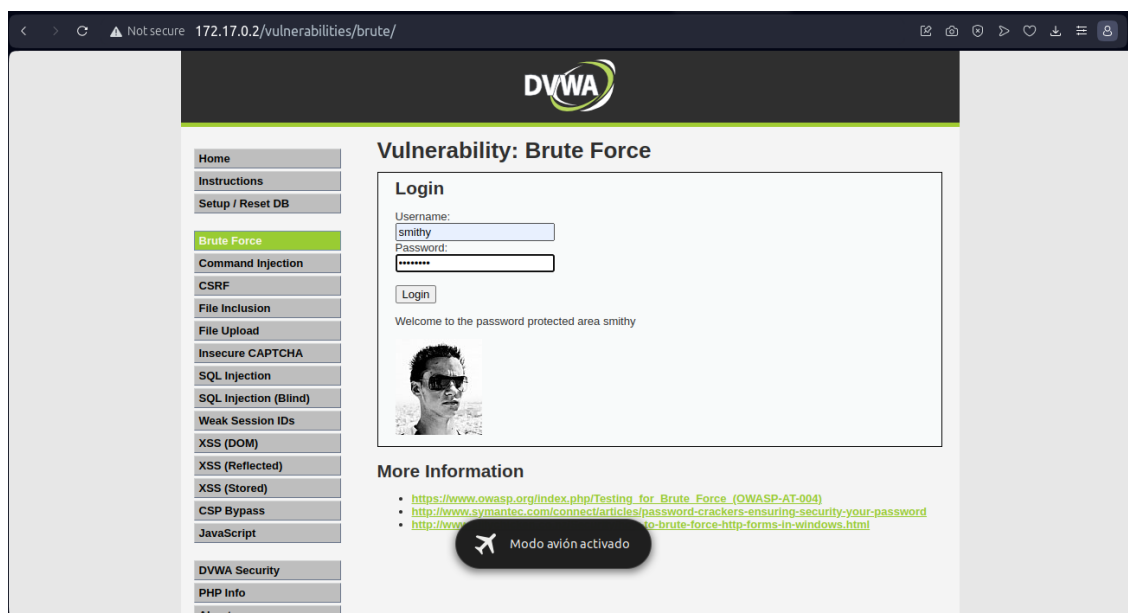


Figura 14: Comprobación número 2

2.7. Obtención de código de inspect element (curl)

En este punto se debe realizar una inspección de elemento al sitio web con el objetivo de determinar el curl a utilizar, al analizar los elementos se logró identificar lo siguiente: Donde se debe realizar una copia de la dirección curl obtenida y de esta manera realizar la solicitud en consola de lo obtenido.

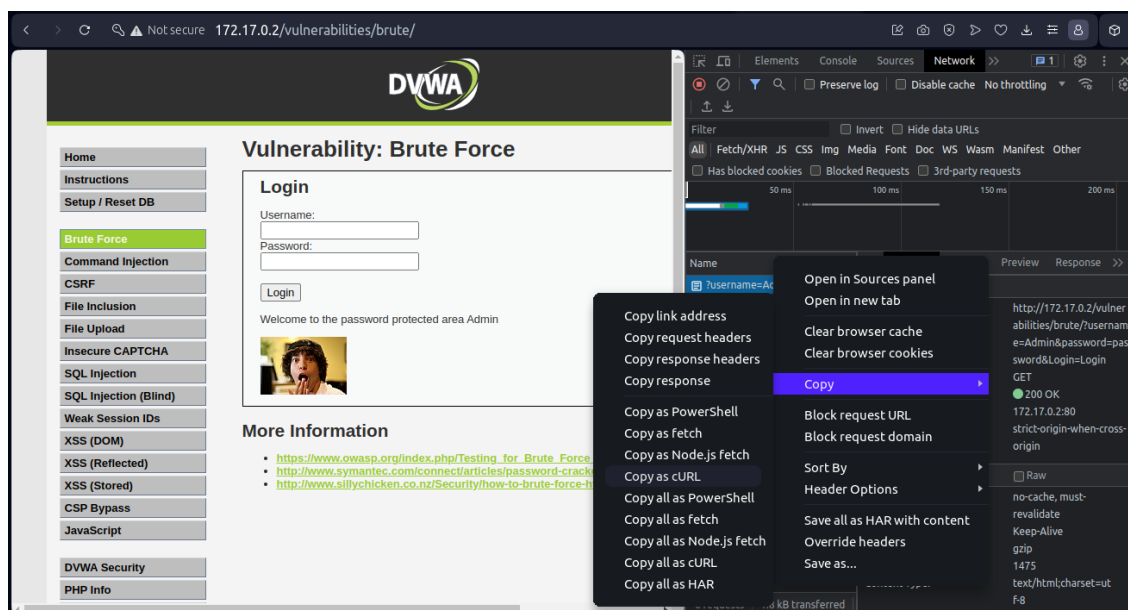


Figura 15: Obtención curl.

2.8. Utilización de curl por terminal (curl)

A partir de lo obtenido anteriormente se realizará la prueba para pares válidos e inválidos con el fin de determinar cuales son las diferencias en las respuestas que se obtienen, la manera en la cuál se realizará lo mencionado es modificar los valores de los campos username y password de manera manual antes de realizar la curl.

Como ya sabemos cuales son usuarios y contraseñas válidos e inválidos se procede a reemplazar, donde se obtuvo lo siguiente:

```

anselm@php-pavillon:~$ curl 'http://172.17.0.2/vulnerabilities/brute/?username=Admin&password=password&login=Login' \
-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=0.7' \
-H 'Accept-Language: es-CL,es;q=0.9,en-US;q=0.8,en;q=0.7,pt-BR;q=0.6,pt;q=0.5' \
-H 'Connection: keep-alive' \
-H 'Cookie: PHPSESSID=ti3bksnkm2qcqlr7p7tv8s814; security=low' \
-H 'Referer: http://172.17.0.2/vulnerabilities/brute/?username=admin&password=ks&login=Login' \
-H 'Upgrade-Insecure-Requests: 1' \
-H 'User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0 Safari/537.36 OPR/102.0.0.0' \
--compressed \
--insecure

<!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="../../../favicon.ico" />
    <script type="text/javascript" src="../../../dvwa/js/dvwaPage.js"></script>
  </head>
  <body class="home">
    <div id="container">
      <div id="header">
        
      </div>
    </div>
  </body>
</html>

```

Figura 16: cURL válido.

```

anselm@php-pavillon:~$ curl 'http://172.17.0.2/vulnerabilities/brute/?username=CR7&password=SIUU&login=Login' \
-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=0.7' \
-H 'Accept-Language: es-CL,es;q=0.9,en-US;q=0.8,en;q=0.7,pt-BR;q=0.6,pt;q=0.5' \
-H 'Connection: keep-alive' \
-H 'Cookie: PHPSESSID=ti3bksnkm2qcqlr7p7tv8s814; security=low' \
-H 'Referer: http://172.17.0.2/vulnerabilities/brute/?username=admin&password=ks&login=Login' \
-H 'Upgrade-Insecure-Requests: 1' \
-H 'User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0 Safari/537.36 OPR/102.0.0.0' \
--compressed \
--insecure

<!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="../../../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">
          <ul class="menuBlocks">
            <li class=""><a href="../../../instructions.php">Instructions</a></li>
            <li class=""><a href="../../../setup.php">Setup / Reset DB</a></li>
          </ul>
        </div>
      </div>
    </div>
  </body>
</html>

```

Figura 17: cURL inválido.

2.9. Demuestra 4 diferencias (curl)

A continuación se deben mencionar 4 diferencias a partir de las cURL realizadas anteriormente.

Para comenzar se debe mencionar la principal diferencia, que corresponde a la respuesta obtenida al momento de realizar la acción por consola.

```
<div class="body_padded">
  <h1>Vulnerability: Brute Forces</h1>

  <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>

  <h2>More Information</h2>
  <ul>
    <li><a href="https://www.owasp.org/index.php/Testing_for_Brute_Force_(OWASP-AT-004)" target="_blank">https://www.owasp.org/index.php/Testing_for_Brute_Force_(OWASP-AT-004)</a></li>
    <li><a href="http://www.synantec.com/connect/articles/password-crackers-ensuring-security-your-password" target="_blank">http://www.synantec.com/connect/articles/password-crackers-ensuring-security-your-password</a></li>
    <li><a href="http://www.sillychicken.co.nz/security/how-to-brute-force-http-forms-in-windows.html" target="_blank">http://www.sillychicken.co.nz/security/how-to-brute-force-http-forms-in-windows.html</a></li>
  </ul>
</div>

<br /><br />

</div>

<div class="clear">
</div>
```

Figura 18: cURL aceptado

```
</ul><ul class="menuBlocks"><li class=""><a href="../../logout.php">Logout</a></li>
</ul>

</div>

</div>

<div id="main_body">

<div class="body_padded">
  <h1>Vulnerability: Brute Forces</h1>

  <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>

  <h2>More Information</h2>
  <ul>
    <li><a href="https://www.owasp.org/index.php/Testing_for_Brute_Force_(OWASP-AT-004)" target="_blank">https://www.owasp.org/index.php/Testing_for_Brute_Force_(OWASP-AT-004)</a></li>
    <li><a href="http://www.synantec.com/connect/articles/password-crackers-ensuring-security-your-password" target="_blank">http://www.synantec.com/connect/articles/password-crackers-ensuring-security-your-password</a></li>
    <li><a href="http://www.sillychicken.co.nz/security/how-to-brute-force-http-forms-in-windows.html" target="_blank">http://www.sillychicken.co.nz/security/how-to-brute-force-http-forms-in-windows.html</a></li>
  </ul>
</div>
```

Figura 19: cURL no aceptado

Otra diferencia que se puede apreciar es el largo del contenido para cada cURL dependiendo su validez.

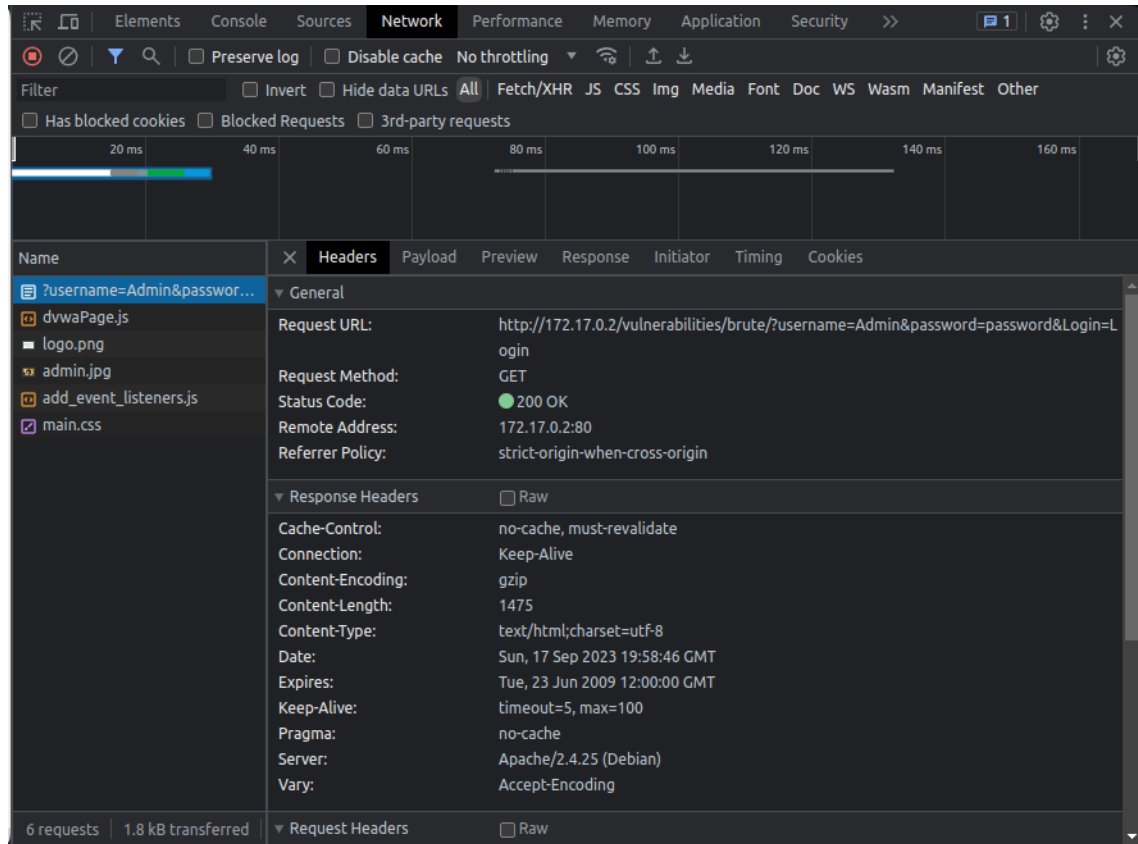


Figura 20: cURL aceptado

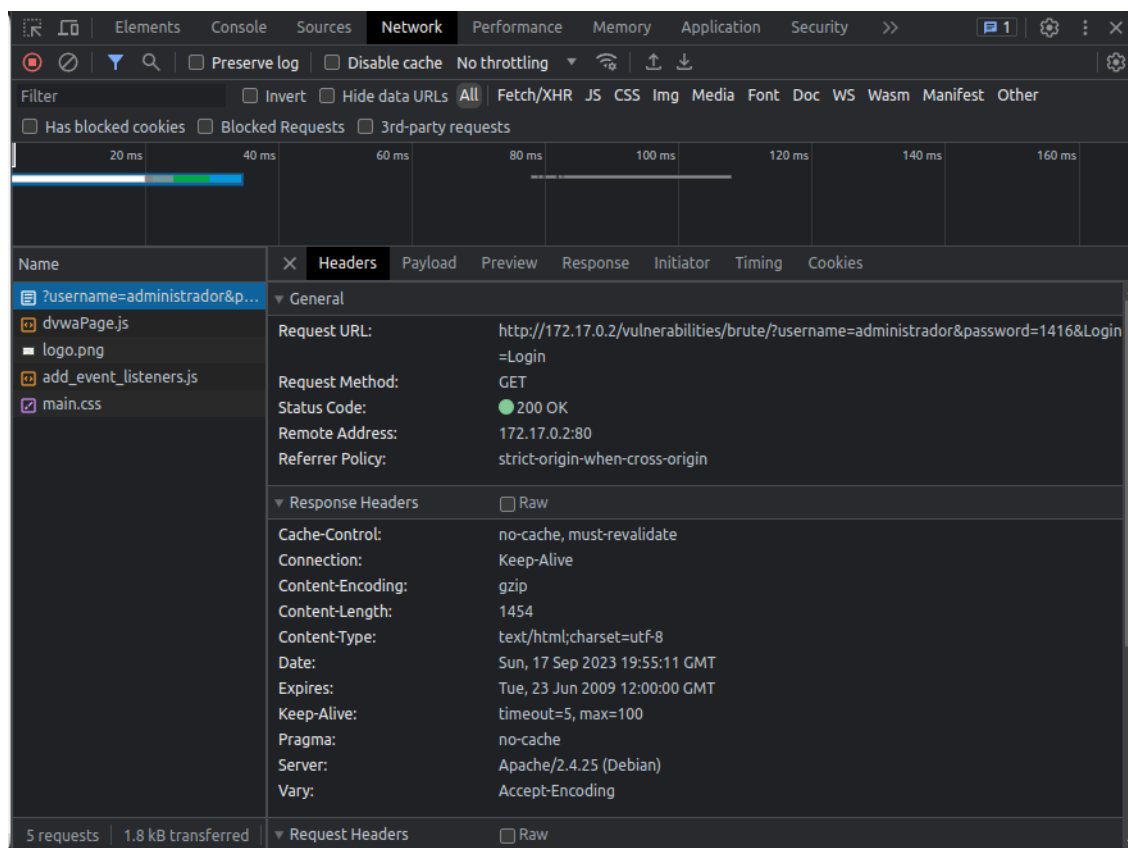


Figura 21: cURL no aceptado

En las imágenes adjuntas anteriormente se puede notar que si se trata de un usuario y contraseña válidos aparece la imagen admin.jpg en las cookies. Lo que permite evidenciar una diferencia entre ambas.

Otra diferencia que puede notarse es que al momento de redireccionar cuando se trata de un usuario es diferente a cuando no lo es. Tal como se aprecia a continuación:

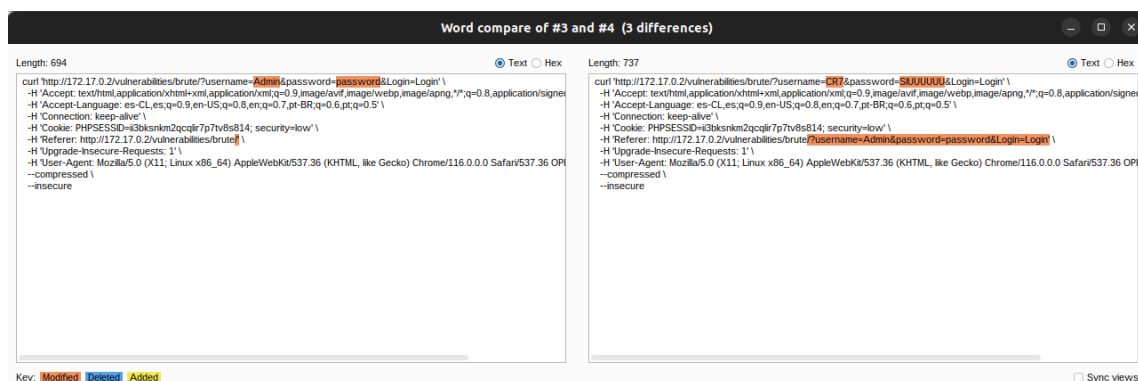


Figura 22: Comparación

Tal como se ve en la imagen se puede notar otra diferencia que sería el largo de caracteres de respuesta a partir de la validez del usuario.

2.10. Instalación y versión a utilizar (hydra)

En este apartado se debe realizar la instalación del software Hydra, la cual se realiza mediante el comando `sudo apt install hydra`.

Para determinar la versión utilizada se utiliza el comando `hydra -version`, el cual permitió determinar que la versión corresponde a Hydra v9.2 (c) 2021 by van Hauser/THC and David Maciejak.

2.11. Explicación de comando a utilizar (hydra)

A continuación se realizará una explicación de la función de cada parte del comando de acuerdo a sus funciones y luego se explicará lo que realiza en conjunto esta herramienta de fuerza bruta.

hydra: Este es el comando principal para ejecutar la herramienta Hydra, que es una herramienta de prueba de fuerza bruta.

172.17.0.2: Esta es la dirección IP del objetivo que se va a atacar.

-m /vulnerabilities/brute/http-get-form.php: Esta opción -m especifica el módulo de autenticación a utilizar. En este caso, se está utilizando un módulo HTTP GET para la autenticación.

http-form-get /vulnerabilities/brute/:username=USER&password=PASS&Login=Login:incorrect PHPSESSID=ii3bksnm2qcqlir7p7tv8s814; security=low": Esta es la cadena que describe la forma en que se va a realizar el ataque. Se indica cómo debe ser la solicitud

HTTP para autenticarse. Explicado detalladamente:

http-form-get: Indica que el método HTTP que se utilizará para la autenticación es GET.

/vulnerabilities/brute/:username=USER&password=PASS&Login= Login:incorrect: H=Cookie: PHPSESSID=ii3bknsnm2qcqlir7p7tv8s814; security=low”: Esta es la cadena de solicitud HTTP que se enviará. Es una URL con los campos username y password, y un campo Login que indica que se está realizando un intento de inicio de sesión. También especifica qué respuesta indica un intento fallido (:incorrect:).

Por último, se incluye una cabecera Cookie que establece ciertas variables de sesión, como PHPSESSID y security.

-L Usuarios.txt: Especifica el archivo Usuarios.txt que contiene la lista de nombres de usuario que se probarán.

-P Contraseñas.txt: Especifica el archivo Contraseñas.txt que contiene la lista de contraseñas que se probarán.

A grandes rasgos este comando se utiliza para realizar un ataque de fuerza bruta utilizando una solicitud HTTP GET contra un objetivo específico (172.17.0.2) que utiliza un formulario de inicio de sesión. Se prueba una lista de nombres de usuario (Usuarios.txt) y una lista de contraseñas (Contraseñas.txt). Se define una solicitud HTTP personalizada y se especifica cómo interpretar las respuestas para determinar si un intento de inicio de sesión fue exitoso o no.

2.12. Obtención de al menos 2 pares (hydra)

Al momento de realizar la ejecución del comando mencionado anteriormente se obtiene la siguiente respuesta, que corresponde a la búsqueda de usuarios y contraseña válidos encontrados en los diccionarios proporcionados para realizar el inicio utilizando fuerza bruta.

```

Hydra v9.2 (c) 2021 by van Hauser/THC & David Maciejak - Please do not use in mi
litary or secret service organizations, or for illegal purposes (this is non-bin
ding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2023-09-17 17:40:
11
[DATA] max 16 tasks per 1 server, overall 16 tasks, 400 login tries (l:20/p:20),
~25 tries per task
[DATA] attacking http-get-form://172.17.0.2:80/vulnerabilities/brute/:username=^
USER^&password=^PASS^&Login=Login:incorrect:H=Cookie: PHPSESSID=ii3bksnkm2qcqlir
7p7tv8s814; security=low
[80][http-get-form] host: 172.17.0.2 login: admin password: password
[80][http-get-form] host: 172.17.0.2 login: smithy password: password
1 of 1 target successfully completed, 2 valid passwords found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2023-09-17 17:40:
17
anselmo@hp-pavilion:~/Descargas$

```

Figura 23: Usuarios registrados

2.13. Explicación paquete curl (tráfico)

Su tráfico puede variar según el comando y la URL que se esté utilizando. No está diseñado específicamente para pruebas de fuerza bruta, por lo que su tráfico puede ser más bajo en comparación con Hydra.

No.	Time	Source	Destination	Protocol	Length	Info
132	13.205725067	172.17.0.1	172.17.0.2	TCP	76	55758 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=35...
133	13.205746090	172.17.0.1	172.17.0.2	TCP	76	[TCP Out-Of-Order] [TCP Port numbers reused] 55758 → 80 [SYN] Seq=0 ...
134	13.205847155	172.17.0.2	172.17.0.1	TCP	76	80 → 55758 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM...
135	13.205858749	172.17.0.2	172.17.0.1	TCP	76	[TCP Out-Of-Order] 80 → 55758 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 ...
136	13.205889760	172.17.0.1	172.17.0.2	TCP	68	55758 → 80 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=3545541225 TSecr=...
137	13.205892623	172.17.0.1	172.17.0.2	TCP	68	[TCP Dup ACK 136#1] 55758 → 80 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSv...
138	13.205991592	172.17.0.1	172.17.0.2	HTTP	766	GET /vulnerabilities/brute/?username=Admin&password=password&login=L...
139	13.205995364	172.17.0.1	172.17.0.2	TCP	766	[TCP Retransmission] 55758 → 80 [PSH, ACK] Seq=1 Ack=1 Win=64256 Len...
140	13.206048934	172.17.0.2	172.17.0.1	TCP	68	80 → 55758 [ACK] Seq=1 Ack=699 Win=64512 Len=0 TSval=3766401528 TSecr...
141	13.206055981	172.17.0.2	172.17.0.1	TCP	68	[TCP Dup ACK 140#1] 80 → 55758 [ACK] Seq=1 Ack=699 Win=64512 Len=0 T...
142	13.211634096	172.17.0.2	172.17.0.1	HTTP	1895	HTTP/1.1 200 OK (text/html)
143	13.211696935	172.17.0.2	172.17.0.1	TCP	1895	[TCP Retransmission] 80 → 55758 [PSH, ACK] Seq=1 Ack=699 Win=64512 L...
144	13.211695629	172.17.0.1	172.17.0.2	TCP	68	55758 → 80 [ACK] Seq=699 Ack=1828 Win=64000 Len=0 TSval=3545541231 T...
145	13.211782888	172.17.0.1	172.17.0.2	TCP	68	[TCP Dup ACK 144#1] 55758 → 80 [ACK] Seq=699 Ack=1828 Win=64000 Len=...

Frame 132: 76 bytes on wire (608 bits), 76 bytes captured (608 bits) on interface any, id 0
 Linux cooked capture v1
 Internet Protocol Version 4, Src: 172.17.0.1, Dst: 172.17.0.2
 Transmission Control Protocol, Src Port: 55758, Dst Port: 80, Seq: 0, Len: 0

0000	00 04 00 01 00 06 02 42	2b 3c 7b 94 00 00 08 00B +c{.....
0010	45 00 00 3c 96 dd 40 00	40 06 4b b9 ac 11 00 01	E<...@_@K.....
0020	ac 11 00 02 d9 ce 00 50	79 14 d4 87 00 00 00 00P y.....
0030	a0 02 fa f0 58 54 00 00	02 04 05 b4 04 02 08 0aXT.....
0040	d3 54 aa 69 00 00 00 00	01 03 03 07	.T.i.....

Figura 24: Tráfico cURL

2.14. Explicación paquete burp (tráfico)

Con esta herramienta se genera tráfico a partir de las solicitudes que se van realizando y a partir de esto se van determinando las respuestas, es decir, para este caso se puede notar que cuando se trata de un usuario y/o contraseña incorrecto se envían paquetes tcp y al ser un par válido se envía la solicitud HTTP.

No.	Time	Source	Destination	Protocol	Length	Info
71	7.990975829	172.17.0.1	172.17.0.2	TCP	76	43344 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=35...
72	7.990996712	172.17.0.1	172.17.0.2	TCP	76	[TCP Out-Of-Order] [TCP Port numbers reused] 43344 → 80 [SYN] Seq=0 ...
73	7.991061039	172.17.0.2	172.17.0.1	TCP	76	80 → 43344 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM...
74	7.991076334	172.17.0.2	172.17.0.1	TCP	76	[TCP Out-Of-Order] 80 → 43344 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0...
75	7.991115447	172.17.0.1	172.17.0.2	TCP	68	43344 → 80 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=3544947495 TSecr=...
76	7.991116760	172.17.0.1	172.17.0.2	TCP	68	[TCP Dup ACK 75#1] 43344 → 80 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSva...
77	7.991466274	172.17.0.1	172.17.0.2	HTTP	671	GET /vulnerabilities/brute/?username=admin&password=pass&login=login...
78	7.991478614	172.17.0.1	172.17.0.2	TCP	671	[TCP Retransmission] 43344 → 80 [PSH, ACK] Seq=1 Ack=1 Win=64256 Len...
79	7.991589269	172.17.0.2	172.17.0.1	TCP	68	80 → 43344 [ACK] Seq=1 Ack=604 Win=64640 Len=0 TSval=3765987798 TSec...
80	7.991605483	172.17.0.2	172.17.0.1	TCP	68	[TCP Dup ACK 79#1] 80 → 43344 [ACK] Seq=1 Ack=604 Win=64640 Len=0 TS...
81	7.99385473	172.17.0.2	172.17.0.1	HTTP	1836	HTTP/1.1 200 OK (text/html)
82	7.999428567	172.17.0.2	172.17.0.1	TCP	1836	[TCP Retransmission] 80 → 43344 [PSH, ACK] Seq=1 Ack=604 Win=64640 L...
83	7.999512240	172.17.0.1	172.17.0.2	TCP	68	43344 → 80 [ACK] Seq=604 Ack=1769 Win=64128 Len=0 TSval=3544947503 T...
84	7.999526628	172.17.0.1	172.17.0.2	TCP	68	[TCP Dup ACK 83#1] 43344 → 80 [ACK] Seq=604 Ack=1769 Win=64128 Len=0...

▶ Frame 71: 76 bytes on wire (608 bits), 76 bytes captured (608 bits) on interface any, id 0
 ▶ Linux cooked capture v1
 ▶ Internet Protocol Version 4, Src: 172.17.0.1, Dst: 172.17.0.2
 ▶ Transmission Control Protocol, Src Port: 43344, Dst Port: 80, Seq: 0, Len: 0

```

0000  00 04 00 01 00 06 02 42 2b 3c 7b 94 00 00 08 00  ....B +{.....
0010  45 00 00 3c 5b c3 40 00 40 06 86 d3 ac 11 00 01  E: <[.: @:.....
0020  ac 11 00 02 a9 50 00 50 97 0a 6d 89 00 00 00 00  ...P.P :m:.....
0030  a0 02 fa f0 58 54 00 00 02 04 05 b4 04 02 08 0a  ...XT.....
0040  d3 4b 9b 27 00 00 00 00 01 03 03 07          K:'.
  
```

Figura 25: Tráfico burp

2.15. Explicación paquete hydra (tráfico)

En esta captura de tráfico se puede apreciar una gran cantidad de tráfico correspondiente al protocolo tcp, que pudiera darse debido a los ataques de fuerza bruta. Es decir, se intentará iniciar sesión repetidamente con diferentes combinaciones de nombres de usuario y contraseñas.

No.	Time	Source	Destination	Protocol	Length	Info
88	15.224950310	172.17.0.1	172.17.0.2	TCP	76	58030 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=35...
89	15.224974057	172.17.0.1	172.17.0.2	TCP	76	[TCP Out-Of-Order] [TCP Port numbers reused] 58030 → 80 [SYN] Seq=0...
90	15.225047323	172.17.0.2	172.17.0.1	TCP	76	80 → 58030 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM...
91	15.225089649	172.17.0.2	172.17.0.1	TCP	76	[TCP Out-Of-Order] 80 → 58030 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0...
92	15.225141683	172.17.0.1	172.17.0.2	TCP	68	58030 → 80 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=3545490636 TSecr=...
93	15.225146432	172.17.0.1	172.17.0.2	TCP	68	[TCP Dup ACK 92#1] 58030 → 80 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSva...
94	15.227575264	172.17.0.1	172.17.0.2	TCP	76	58044 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=35...
95	15.227601386	172.17.0.1	172.17.0.2	TCP	76	[TCP Out-Of-Order] [TCP Port numbers reused] 58044 → 80 [SYN] Seq=0...
96	15.227667808	172.17.0.2	172.17.0.1	TCP	76	80 → 58044 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM...
97	15.227683313	172.17.0.2	172.17.0.1	TCP	76	[TCP Out-Of-Order] 80 → 58044 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0...
98	15.227726407	172.17.0.1	172.17.0.2	TCP	68	58044 → 80 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=3545490639 TSecr=...
99	15.227729271	172.17.0.1	172.17.0.2	TCP	68	[TCP Dup ACK 98#1] 58044 → 80 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSva...
100	15.229849880	172.17.0.1	172.17.0.2	TCP	76	58054 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=35...
101	15.229862732	172.17.0.1	172.17.0.2	TCP	76	[TCP Out-Of-Order] [TCP Port numbers reused] 58054 → 80 [SYN] Seq=0...

▶ Frame 88: 76 bytes on wire (608 bits), 76 bytes captured (608 bits) on interface any, id 0
 ▶ Linux cooked capture v1
 ▶ Internet Protocol Version 4, Src: 172.17.0.1, Dst: 172.17.0.2
 ▶ Transmission Control Protocol, Src Port: 58030, Dst Port: 80, Seq: 0, Len: 0

```

0000  00 04 00 01 00 06 02 42 2b 3c 7b 94 00 00 08 00  .....B +{.....
0010  45 00 00 3c b5 f1 40 00 40 06 2c a5 ac 11 00 01  E<...@...
0020  ac 11 00 02 e2 ae 00 50 52 d7 49 0e 00 00 00 00  ...P R I...
0030  a0 02 fa f0 58 54 00 00 02 04 05 b4 04 02 08 0a  ...XT...
0040  d3 53 e4 cc 00 00 00 00 01 03 03 07          S.....
  
```

Figura 26: Tráfico hydra

2.16. Mención de las diferencias (tráfico)

Las diferencia que se pudieron notar entre las herramientas es la cantidad de paquetes generados a partir de la ejecución de las actividades realizadas, debido a que en hydra se generaron una mayor cantidad de paquetes en comparación a las otras herramientas, principalmente del protocolo TCP.

Lo que puede producir diferencias entre estas herramientas son las configuraciones y opciones específicas que se utilicen en cada herramienta, ya que estas pueden influir en la cantidad y el tipo de tráfico generado.

2.17. Detección de SW (tráfico)

No.	Time	Source	Destination	Protocol	Length	Info
10408	235.188744303	192.168.43.111	239.255.255.250	SSDP	210	M-SEARCH * HTTP/1.1
10409	235.744770393	192.168.43.111	239.255.255.250	SSDP	207	M-SEARCH * HTTP/1.1
10410	236.189581377	192.168.43.111	239.255.255.250	SSDP	210	M-SEARCH * HTTP/1.1
10411	236.193030927	192.168.43.111	35.186.224.42	TLSv1.2	109	Application Data
10412	236.227212651	35.186.224.42	192.168.43.111	TCP	66	443 → 46314 [ACK] Seq=281 Ack=345 Win=268 Len=0 TSval=1461412347...
10413	237.012263047	35.186.224.42	192.168.43.111	TLSv1.2	106	Application Data
10414	237.012368873	192.168.43.111	35.186.224.42	TCP	66	46314 → 443 [ACK] Seq=345 Ack=321 Win=501 Len=0 TSval=2978777772...
10415	237.820222364	192.168.43.111	185.199.108.154	TCP	66	[TCP Dup ACK 41#5] 39164 → 443 [ACK] Seq=1 Ack=1 Win=1695 Len=0 ...
10416	237.876321617	185.199.108.154	192.168.43.111	TCP	66	[TCP Dup ACK 42#5] [TCP ACKed unseen segment] 443 → 39164 [ACK] ...
10419	241.372245372	192.168.43.111	185.199.108.154	TCP	66	[TCP Dup ACK 87#5] 33034 → 443 [ACK] Seq=1 Ack=1 Win=700 Len=0 T...
10420	241.372501142	192.168.43.111	185.199.108.133	TCP	66	[TCP Dup ACK 89#5] 41710 → 443 [ACK] Seq=1 Ack=1 Win=501 Len=0 T...
Frame 3: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface wlo1, id 0						
Ethernet II, Src: 22:32:6c:2b:eb:2b (22:32:6c:2b:eb:2b), Dst: HonHaiPr_d0:3e:49 (48:e2:44:d0:3e:49)						
Internet Protocol Version 4, Src: 35.186.224.19, Dst: 192.168.43.111						
Transmission Control Protocol, Src Port: 443, Dst Port: 37622, Seq: 1, Ack: 1, Len: 0						
0000	48 e2 44 d0 3e 49 22 32	6c 2b eb 2b 08 00 45 68	H.D.>I"2 l:~...Eh			
0010	00 34 00 32 00 00 7a 06	49 45 23 ba e0 13 c0 a8	4-2~Z HE#....			
0020	2b ef 01 bb 92 f6 82 5a	7b 5c f4 a0 04 99 89 11	+0~...Z (\.....			
0030	01 05 f6 08 00 00 01 01	08 0a c9 33 21 1c 07 44	~.....~3!~D			
0040	12 93		..			

Figura 27: Tráfico SW

2.18. Interacción con el formulario (python)

2.19. Cabeceras HTTP (python)

2.20. Obtención de al menos 2 pares (python)

2.21. Comparación de rendimiento con Hydra, Burpsuite, y cURL (python)

2.22. Demuestra 4 métodos de mitigación (investigación)

- **Limitación de intentos de inicio de sesión:** Este método consiste en restringir el número de intentos fallidos de inicio de sesión permitidos en un período de tiempo. Si un usuario excede un número determinado de intentos fallidos (por ejemplo, 5 intentos), la cuenta puede ser bloqueada temporalmente o se requiere verificación adicional (como un CAPTCHA o autenticación por correo electrónico). Esto dificulta que los atacantes puedan realizar un gran número de intentos en poco tiempo.
- **Autenticación multifactor (MFA):** La autenticación multifactor requiere que los usuarios verifiquen su identidad mediante múltiples formas de autenticación (algo que saben, como una contraseña, y algo que tienen, como un token de autenticación o un código enviado por SMS). Esto agrega una capa adicional de seguridad, ya que incluso si un atacante logra obtener las credenciales, necesitaría el segundo factor para acceder a la cuenta.
- **CAPTCHA:** Los CAPTCHA (Completely Automated Public Turing test to tell Computers and Humans Apart) se utilizan para verificar que el intento de acceso está siendo realizado por un humano y no por un bot automatizado. Se pueden implementar después de un cierto número de intentos fallidos de inicio de sesión para prevenir ataques automatizados de fuerza bruta. Los CAPTCHA pueden ser tradicionales (con imágenes o texto) o invisibles (detectar comportamiento humano en el sitio).
- **Hashing y salting de contraseñas:** Almacenar contraseñas de manera segura es crucial para prevenir ataques de fuerza bruta offline. Las contraseñas deben ser *hasheadas* usando algoritmos robustos como bcrypt o Argon2, y acompañadas de un *sal* (un valor aleatorio añadido a la contraseña antes de hashearla). Esto hace que incluso si un atacante obtiene acceso a la base de datos, las contraseñas en texto claro no sean accesibles, y un ataque de fuerza bruta sea menos efectivo debido al costo computacional que implica hashear cada posible combinación.

Conclusiones y comentarios

De esta actividad se puede concluir que cada herramienta genera tráfico de manera diferente debido a su funcionalidad y propósito. Por ejemplo, Hydra generará tráfico de fuerza bruta intenso, mientras que cURL y Burp Suite están diseñados para propósitos más específicos. Como es el caso de burp suite que se utiliza para protocolo Http.

Por otra parte se puede notar que las técnicas de fuerza bruta son poderosas. En este caso fueron útiles para probar la robustez de contraseñas y que los HTTP se pueden vulnerar de manera sencilla utilizando las herramientas y técnicas empleadas en esta experiencia.