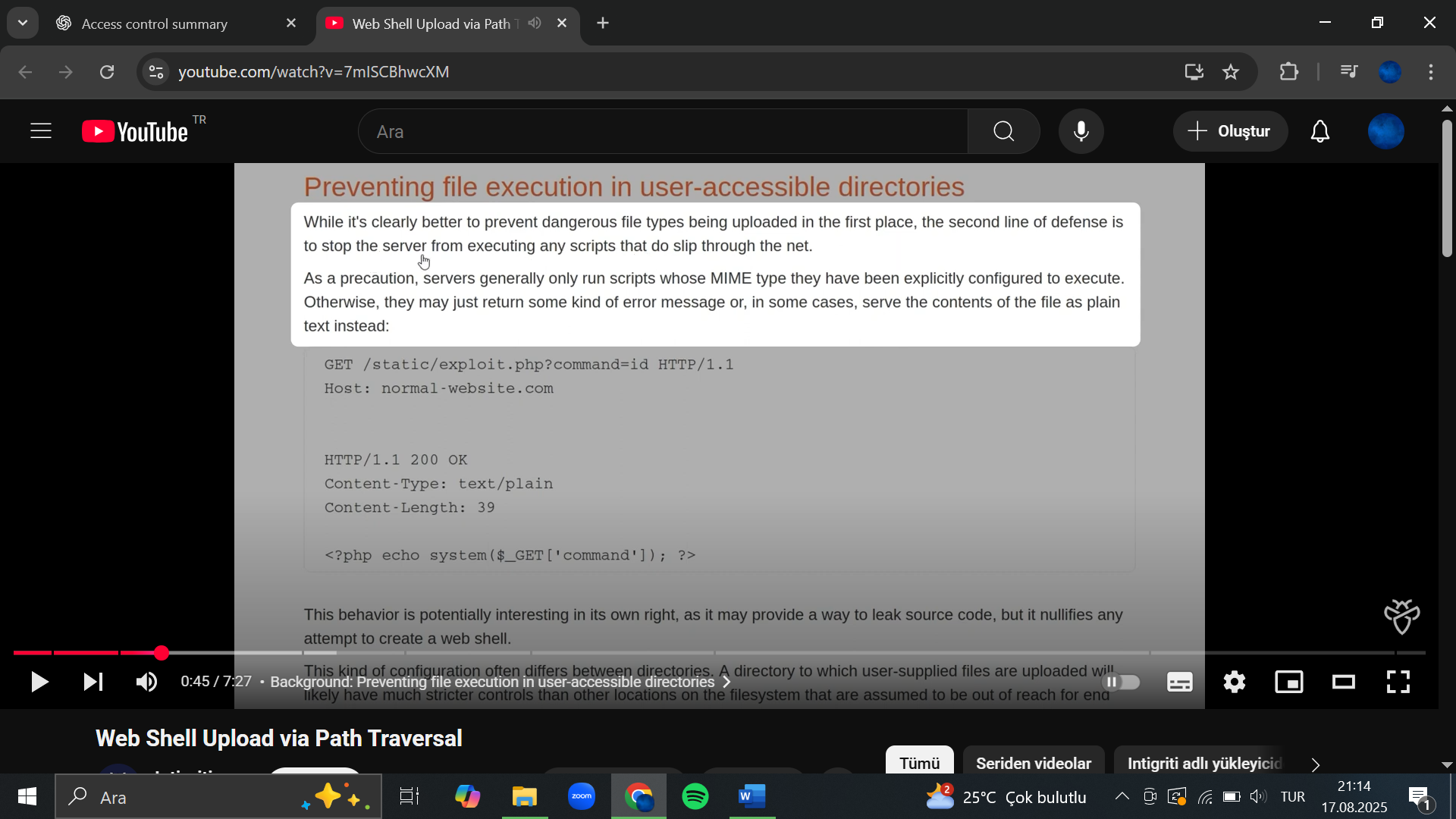
# PortSwigger File Upload Vulnerabilities Lab Notes



# Remote code execution via web shell upload

This lab contains a vulnerable image upload function. It doesn't perform any validation on the files users upload before storing them on the server's filesystem.

To solve the lab, upload a basic PHP web shell and use it to exfiltrate the contents of the file /home/carlos/secret. Submit this secret using the button provided in the lab banner.

You can log in to your own account using the following credentials: wiener:peter

 Solution

1. While proxying traffic through Burp, log in to your account and notice the option for uploading an avatar image.
2. Upload an arbitrary image, then return to your account page. Notice that a preview of your avatar is now displayed on the page.
3. In Burp, go to **Proxy > HTTP history**. Click the filter bar to open the **HTTP history filter** window. Under **Filter by MIME type**, enable the **Images** checkbox, then apply your changes.
4. In the proxy history, notice that your image was fetched using a GET request to /files/avatars/<YOUR-IMAGE>. Send this request to Burp Repeater.
5. On your system, create a file called exploit.php, containing a script for fetching the contents of Carlos's secret file. For example:

***<?php echo file\_get\_contents('/home/carlos/secret'); ?>***

1. Use the avatar upload function to upload your malicious PHP file. The message in the response confirms that this was uploaded successfully.
2. In Burp Repeater, change the path of the request to point to your PHP file:

***GET /files/avatars/exploit.php HTTP/1.1***

1. Send the request. Notice that the server has executed your script and returned its output (Carlos's secret) in the response.
2. Submit the secret to solve the lab.
3. Web shell upload via Content-Type restriction bypass

This lab contains a vulnerable image upload function. It attempts to prevent users from uploading unexpected file types, but relies on checking user-controllable input to verify this.

To solve the lab, upload a basic PHP web shell and use it to exfiltrate the contents of the file /home/carlos/secret. Submit this secret using the button provided in the lab banner.

You can log in to your own account using the following credentials: wiener:peter

 Solution

1. Log in and upload an image as your avatar, then go back to your account page.
2. In Burp, go to **Proxy > HTTP history** and notice that your image was fetched using a GET request to /files/avatars/<YOUR-IMAGE>. Send this request to Burp Repeater.
3. On your system, create a file called exploit.php, containing a script for fetching the contents of Carlos's secret. For example:

***<?php echo file\_get\_contents('/home/carlos/secret'); ?>***

1. Attempt to upload this script as your avatar. The response indicates that you are only allowed to upload files with the MIME type image/jpeg or image/png.
2. In Burp, go back to the proxy history and find the POST /my-account/avatar request that was used to submit the file upload. Send this to Burp Repeater.
3. In Burp Repeater, go to the tab containing the POST /my-account/avatar request. In the part of the message body related to your file, change the specified Content-Type to image/jpeg.
4. Send the request. Observe that the response indicates that your file was successfully uploaded.
5. Switch to the other Repeater tab containing the GET /files/avatars/<YOUR-IMAGE> request. In the path, replace the name of your image file with exploit.php and send the request. Observe that Carlos's secret was returned in the response.
6. Submit the secret to solve the lab.
7. Web shell upload via path traversal

This lab contains a vulnerable image upload function. The server is configured to prevent execution of user-supplied files, but this restriction can be bypassed by exploiting a [secondary vulnerability](https://portswigger.net/web-security/file-path-traversal).

To solve the lab, upload a basic PHP web shell and use it to exfiltrate the contents of the file /home/carlos/secret. Submit this secret using the button provided in the lab banner.

You can log in to your own account using the following credentials: wiener:peter

 Solution

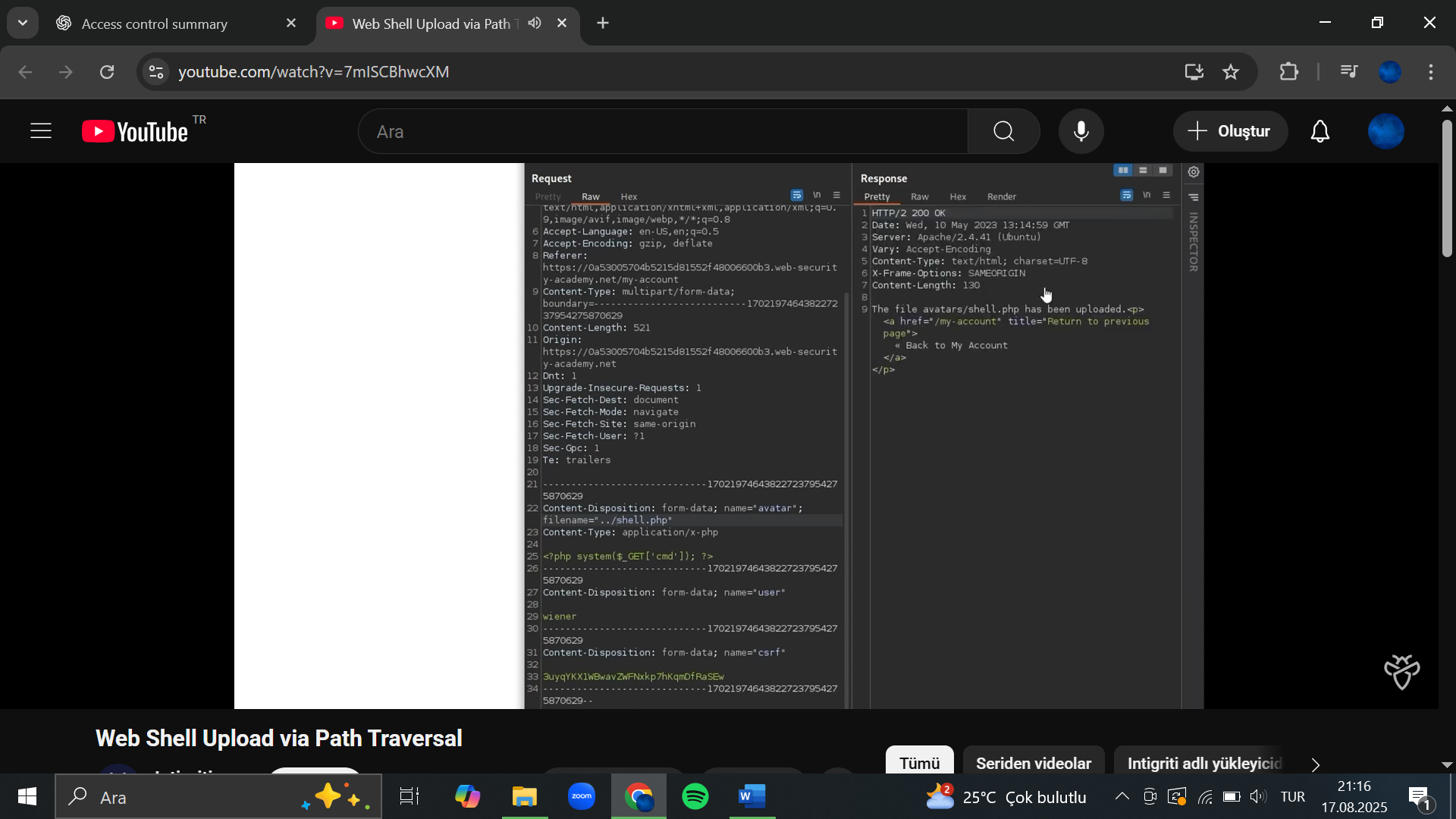
1. Log in and upload an image as your avatar, then go back to your account page.
2. In Burp, go to **Proxy > HTTP history** and notice that your image was fetched using a GET request to /files/avatars/<YOUR-IMAGE>. Send this request to Burp Repeater.
3. On your system, create a file called exploit.php, containing a script for fetching the contents of Carlos's secret. For example:

***<?php echo file\_get\_contents('/home/carlos/secret'); ?>***

1. Upload this script as your avatar. Notice that the website doesn't seem to prevent you from uploading PHP files.
2. In Burp Repeater, go to the tab containing the GET /files/avatars/<YOUR-IMAGE> request. In the path, replace the name of your image file with exploit.php and send the request. Observe that instead of executing the script and returning the output, the server has just returned the contents of the PHP file as plain text.
3. In Burp's proxy history, find the POST /my-account/avatar request that was used to submit the file upload and send it to Burp Repeater.
4. In Burp Repeater, go to the tab containing the POST /my-account/avatar request and find the part of the request body that relates to your PHP file. In the Content-Disposition header, change the filename to include a directory traversal sequence:

***Content-Disposition: form-data; name="avatar"; filename="../exploit.php"***

1. Send the request. Notice that the response says The file avatars/exploit.php has been uploaded. This suggests that the server is stripping the directory traversal sequence from the file name.



1. Obfuscate the directory traversal sequence by URL encoding the forward slash (/) character, resulting in:

***filename="..%2fexploit.php"***

1. Send the request and observe that the message now says The file avatars/../exploit.php has been uploaded. This indicates that the file name is being URL decoded by the server.

A screenshot of a computer

AI-generated content may be incorrect.

1. In the browser, go back to your account page.
2. In Burp's proxy history, find the ***GET /files/avatars/..%2fexploit.php*** request. Observe that Carlos's secret was returned in the response. This indicates that the file was uploaded to a higher directory in the filesystem hierarchy (/files), and subsequently executed by the server. Note that this means you can also request this file using GET /files/exploit.php.

A screenshot of a computer

AI-generated content may be incorrect.

1. Submit the secret to solve the lab.
2. Web shell upload via extension blacklist bypass

This lab contains a vulnerable image upload function. Certain file extensions are blacklisted, but this defense can be bypassed due to a fundamental flaw in the configuration of this blacklist.

To solve the lab, upload a basic PHP web shell, then use it to exfiltrate the contents of the file /home/carlos/secret. Submit this secret using the button provided in the lab banner.

You can log in to your own account using the following credentials: wiener:peter

**Hint**

You need to upload two different files to solve this lab.

 Solution

1. Log in and upload an image as your avatar, then go back to your account page.
2. In Burp, go to **Proxy > HTTP history** and notice that your image was fetched using a GET request to /files/avatars/<YOUR-IMAGE>. Send this request to Burp Repeater.
3. On your system, create a file called exploit.php containing a script for fetching the contents of Carlos's secret. For example:

***<?php echo file\_get\_contents('/home/carlos/secret'); ?>***

1. Attempt to upload this script as your avatar. The response indicates that you are not allowed to upload files with a .php extension.
2. In Burp's proxy history, find the POST /my-account/avatar request that was used to submit the file upload. In the response, notice that the headers reveal that you're talking to an Apache server. Send this request to Burp Repeater.
3. In Burp Repeater, go to the tab for the POST /my-account/avatar request and find the part of the body that relates to your PHP file. Make the following changes:
   * ***Change the value of the filename parameter to .htaccess.***
   * ***Change the value of the Content-Type header to text/plain.***
   * ***Replace the contents of the file (your PHP payload) with the following Apache directive:***

***AddType application/x-httpd-php .l33t***

This maps an arbitrary extension (.l33t) to the executable MIME type application/x-httpd-php. As the server uses the mod\_php module, it knows how to handle this already.

1. Send the request and observe that the file was successfully uploaded.
2. Use the back arrow in Burp Repeater to return to the original request for uploading your PHP exploit.
3. Change the value of the filename parameter from exploit.php to exploit.l33t. Send the request again and notice that the file was uploaded successfully.
4. Switch to the other Repeater tab containing the GET /files/avatars/<YOUR-IMAGE> request. In the path, replace the name of your image file with exploit.l33t and send the request. Observe that Carlos's secret was returned in the response. Thanks to our malicious .htaccess file, the .l33t file was executed as if it were a .php file.
5. Submit the secret to solve the lab.
6. Web shell upload via obfuscated file extension

This lab contains a vulnerable image upload function. Certain file extensions are blacklisted, but this defense can be bypassed using a classic obfuscation technique.

To solve the lab, upload a basic PHP web shell, then use it to exfiltrate the contents of the file /home/carlos/secret. Submit this secret using the button provided in the lab banner.

You can log in to your own account using the following credentials: wiener:peter

 Solution

1. Log in and upload an image as your avatar, then go back to your account page.
2. In Burp, go to **Proxy > HTTP history** and notice that your image was fetched using a GET request to /files/avatars/<YOUR-IMAGE>. Send this request to Burp Repeater.
3. On your system, create a file called exploit.php, containing a script for fetching the contents of Carlos's secret. For example:

***<?php echo file\_get\_contents('/home/carlos/secret'); ?>***

1. Attempt to upload this script as your avatar. The response indicates that you are only allowed to upload JPG and PNG files.
2. In Burp's proxy history, find the POST /my-account/avatar request that was used to submit the file upload. Send this to Burp Repeater.
3. In Burp Repeater, go to the tab for the POST /my-account/avatar request and find the part of the body that relates to your PHP file. In the ***Content-Disposition*** header, change the value of the filename parameter to include a URL encoded null byte, followed by the .jpg extension:

***filename="exploit.php%00.jpg"***

1. Send the request and observe that the file was successfully uploaded. Notice that the message refers to the file as exploit.php, suggesting that the null byte and .jpg extension have been stripped.
2. Switch to the other Repeater tab containing the GET /files/avatars/<YOUR-IMAGE> request. In the path, replace the name of your image file with exploit.php and send the request. Observe that Carlos's secret was returned in the response.
3. Submit the secret to solve the lab
4. Remote code execution via polyglot web shell upload

This lab contains a vulnerable image upload function. Although it checks the contents of the file to verify that it is a genuine image, it is still possible to upload and execute server-side code.

To solve the lab, upload a basic PHP web shell, then use it to exfiltrate the contents of the file /home/carlos/secret. Submit this secret using the button provided in the lab banner.

You can log in to your own account using the following credentials: wiener:peter

 Solution

1. On your system, create a file called exploit.php containing a script for fetching the contents of Carlos's secret. For example:

***<?php echo file\_get\_contents('/home/carlos/secret'); ?>***

1. Log in and attempt to upload the script as your avatar. Observe that the server successfully blocks you from uploading files that aren't images, even if you try using some of the techniques you've learned in previous labs.
2. Create a polyglot PHP/JPG file that is fundamentally a normal image, but contains your PHP payload in its metadata. A simple way of doing this is to download and run ExifTool from the command line as follows:

***exiftool -Comment="<?php echo 'START ' . file\_get\_contents('/home/carlos/secret') . ' END'; ?>" <YOUR-INPUT-IMAGE>.jpg -o polyglot.php***

This adds your PHP payload to the image's Comment field, then saves the image with a .php extension.

1. In the browser, upload the polyglot image as your avatar, then go back to your account page.
2. In Burp's proxy history, find the GET /files/avatars/polyglot.php request. Use the message editor's search feature to find the START string somewhere within the binary image data in the response. Between this and the END string, you should see Carlos's secret, for example:

***START 2B2tlPyJQfJDynyKME5D02Cw0ouydMpZ END***

1. Submit the secret to solve the lab.