









# PERISAI FINAL WRITE UP

## WEB

### BlackHole

This challenge reminds of the Bring Your Own Script challenge in the preliminary round where it has A LOT of directories

### Index of /

<u>Name</u>	<u>Last modified</u>	<u>Size</u>	<u>Description</u>
 <a href="#">0c/</a>	2024-03-03 09:55	-	
 <a href="#">0d/</a>	2024-03-03 09:55	-	
 <a href="#">00/</a>	2024-03-03 09:55	-	
 <a href="#">02/</a>	2024-03-03 09:55	-	
 <a href="#">04/</a>	2024-03-03 09:55	-	
 <a href="#">05/</a>	2024-03-03 09:55	-	
 <a href="#">07/</a>	2024-03-03 09:55	-	
 <a href="#">09/</a>	2024-03-03 09:55	-	

So my first thought is to crawl all the endpoints with GoSpider and with the word flag to narrow down my search to see what is available.

```
└─(j4de@kali)-[~]
└─$ gospider -s 'https://blackhole.ctf.rawsec.com/' -d 0 --robots | grep flag
[robots] - https://blackhole.ctf.rawsec.com/flag-generator.php
[url] - [code-200] - https://blackhole.ctf.rawsec.com/not_flag_do_not_open.txt
[url] - [code-200] - https://blackhole.ctf.rawsec.com/flag-generator.php
[url] - [code-200] - https://blackhole.ctf.rawsec.com/not_flag.txt
[url] - [code-200] - https://blackhole.ctf.rawsec.com/52/57/53/43/7b/62/6c/34/63/6b/68/30/6c/65/
```

Then i stumbled upon

```
"https://blackhole.ctf.rawsec.com/52/57/53/43/7b/62/6c/34/63/6b/68/30/6c/65/5f/69/7a/5f/77/30/72/6d/68/30/6c/33/7d/f
```

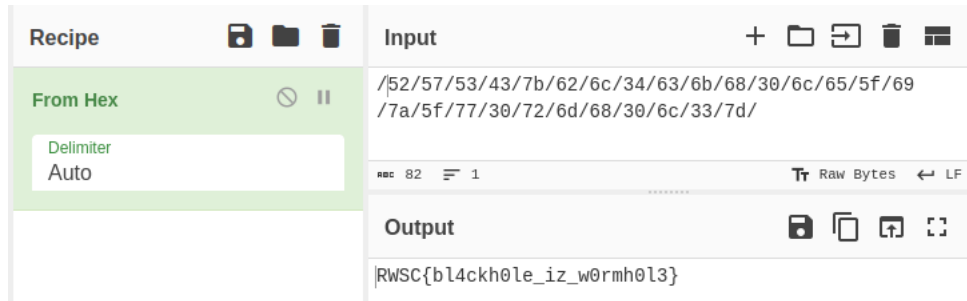
```
└─(j4de@kali)-[~]
└─$ curl https://blackhole.ctf.rawsec.com/52/57/53/43/7b/62/6c/34/63/6b/68/30/6c/65/5f/69/7a/5f/
are you finding the flag?
```

this is not a flag.

check back the url!

Check back the URL huh? The directories leading to the flag.txt looks sus. So when i checked the URL

"/52/57/53/43/7b/62/6c/34/63/6b/68/30/6c/65/5f/69/7a/5f/77/30/72/6d/68/30/6c/33/7d", it looks like a hex string and i input it into CyberChef from hex and got the flag



## Anti-Brute

Oh no, I forgot the password for my login page. Can you help me to find the password for my admin user. Retrieve it and submit as RWSC{flag}.

<https://no-brute.ctf.rawsec.com/>

The challenge asks us to get the password of the admin user. They also provided a wordlist of possible passwords. So i scripted a python code to bruteforce the the login page and present the contents if the contents are not "Invalid username or password".

```
import requests

def try_password(password):
    url = "https://no-brute.ctf.rawsec.com/login.php"
    data = {"username": "admin", "password": password}

    response = requests.post(url, data=data)

    if "Invalid username or password" not in response.text:
        print(f"Password found: {password}")
        print(response.text)
        return True
    else:
```

```

        return False

# Replace 'path/to/wordlist.txt' with the actual path to your wordlist file
wordlist_path = 'pass.txt'

with open(wordlist_path, 'r') as wordlist_file:
    for line in wordlist_file:
        password = line.strip()
        if try_password(password):
            break

```

Upon running the script, we got the flag

```

└─(j4de@kali) - [~/CTF/RENTASFinal/Web/Anti-Brute]
└─$ /bin/python /home/j4de/CTF/RENTASFinal/Web/Anti-Brute/brute.py
Password found: secretpass
<p class="congratulations"><strong>Congratulations! Here your flag: RWSC{n0_brut3f0rc3_pl34s3}</

```

## MISC

### Motivation

Upon extracting the rar file, we have so many qrcodes. Also based on the hint, it says "Dig Everything, might be somewhere in "name"?". So my initial thought was to get all the names of the files into a single text file

```

└─(j4de@kali) - [~/CTF/RENTASFinal/MISC/some-motivation]
└─$ ls > name.txt

```

Then i removed the .png extention using vim to get only the file names:

```

:%s/\.png/d

```

The file name looks like it is in hex format:

```

└─(j4de@kali) - [~/CTF/RENTASFinal/MISC/some-motivation]
└─$ cat name.txt | head
001229d145fd784c8b2bfed0ed947f6ad
009b639d4dd46c9e5ca45dda6a4709f3d
00e85061b089b78c618689afa3413a56d
00e98f8a6e5e87e5a55d7a850b4fad42d
00f2dcb9fece8d7e0b9a37e5b48d9895d
0119504bba4a9f35a071af65554de0acd
01521e77dc3845c76613a9f999ee130bd
018f240c59c543913d3062c374292002d
0199e590e506516e1bcd87b291a7b2ad
01be6cbbad0b46d3544c51d25b0970a5d

```

So i decided to upload the file into cyberchef, decode it from hex and searched for the flag format.



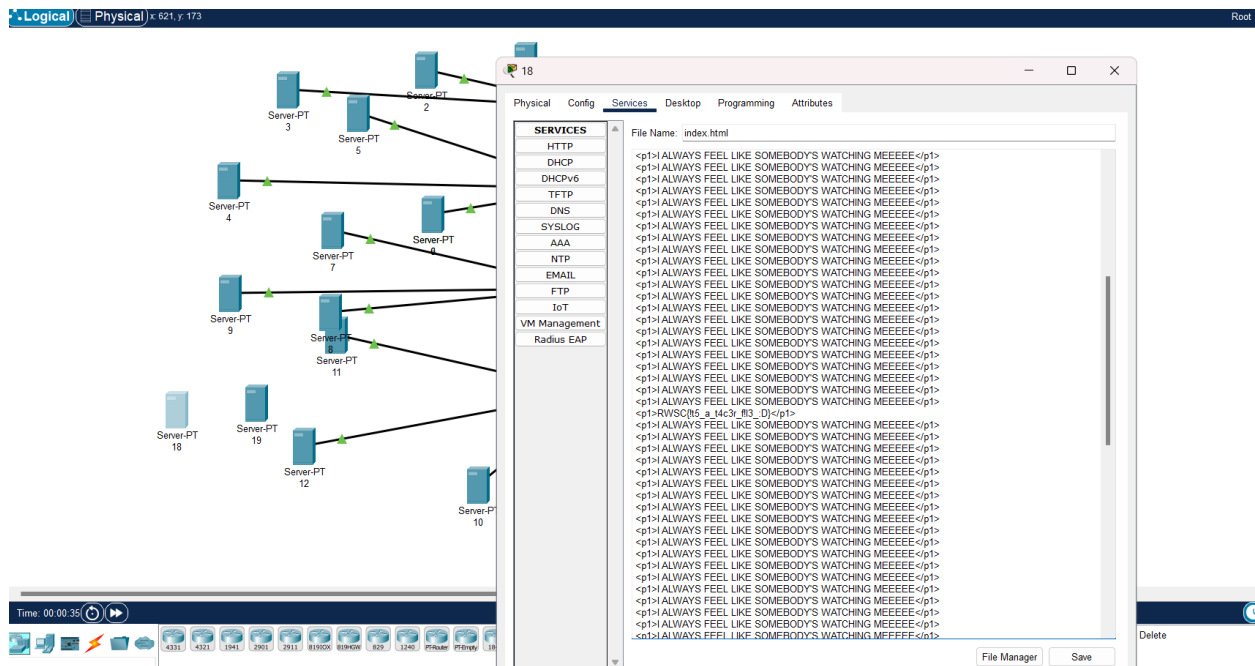
- 2s light indicating .
- >2s light indicating -

and decode it using [online morse code decoder](#)

## NETWORK

### I Hope You Have The Software

We are given a PKT file which can be open using Cisco Packet Tracer . At first, we try to find any weird things in each server but after one of my teammate realised that there is other server that is stacked to each other, I found one server that is interesting and got the flag in it.



## DFIR

### Hiding Zombie

So we are given a PNG file and there is something wrong with the PNG file.

First step is, we tried to use all tools like exiftool, pngcheck, and some online tools to check if there is any hidden flags in the image. Referring to this website [https://hackmd.io/@FlsYpINBRKixPQQVbh98kw/Sk\\_IVRCBr](https://hackmd.io/@FlsYpINBRKixPQQVbh98kw/Sk_IVRCBr), we tried to repair the image file.

From the website, what we learnt is, after we got an error from pngcheck, we need to change the CRC value in IHDR chunks

So after fixing the file, it shows a zombie holding a flag, but there is no flag found in the image. Then we ran exiftool to view the metadata of the image and found out that the pixels per unit for X and Y is not tally with the image size. (It should be the same)

```
(j4de@kali) - [~/CTF/RENTASFinal/DFIR]
$ exiftool zomba2.png
ExifTool Version Number      : 12.65
File Name                    : zomba2.png
Directory                   : .
File Size                    : 1537 kB
File Modification Date/Time  : 2024:03:09 16:52:27+08:00
File Access Date/Time       : 2024:03:09 16:52:28+08:00
File Inode Change Date/Time  : 2024:03:09 16:52:28+08:00
File Permissions             : -rw-r--r--
File Type                    : PNG
File Type Extension          : png
MIME Type                    : image/png
```

```

Image Width      : 1024
Image Height     : 921
Bit Depth        : 8
Color Type       : RGB with Alpha
Compression      : Deflate/Inflate
Filter           : Adaptive
Interlace        : Noninterlaced
SRGB Rendering   : Perceptual
Gamma            : 2.2
Pixels Per Unit X : 3780
Pixels Per Unit Y : 3780
Pixel Units      : meters
Image Size       : 1024x921
Megapixels       : 0.943

```

So we decided to view the hex via hexedit to change the values of the height from 03 to 04 0x00000016. However, the file became corrupted again and we did the same step as before this with png check.

```

(j4de@kali)-[~/CTF/RENTASFinal/DFIR]
└─$ pngcheck -v zombaokhex.png
File: zombaokhex.png (1536669 bytes)
  chunk IHDR at offset 0x0000c, length 13
    1024 x 1177 image, 32-bit RGB+alpha, non-interlaced
  CRC error in chunk IHDR (computed 45746c08, expected 80d35286)
ERRORS DETECTED in zombaokhex.png

```

The first screenshot shows the original hex data for the IHDR chunk. The height value is 03 (at offset 00000010). The CRC error is reported as computed 45746c08, expected 80d35286.

The second screenshot shows the modified hex data. The height value has been changed to 04 (at offset 00000010). The CRC error is now reported as computed 80d35286, expected 45746c08.

So we change the values from 0x0000001D to 0x00000020 as the image and we got the flag

