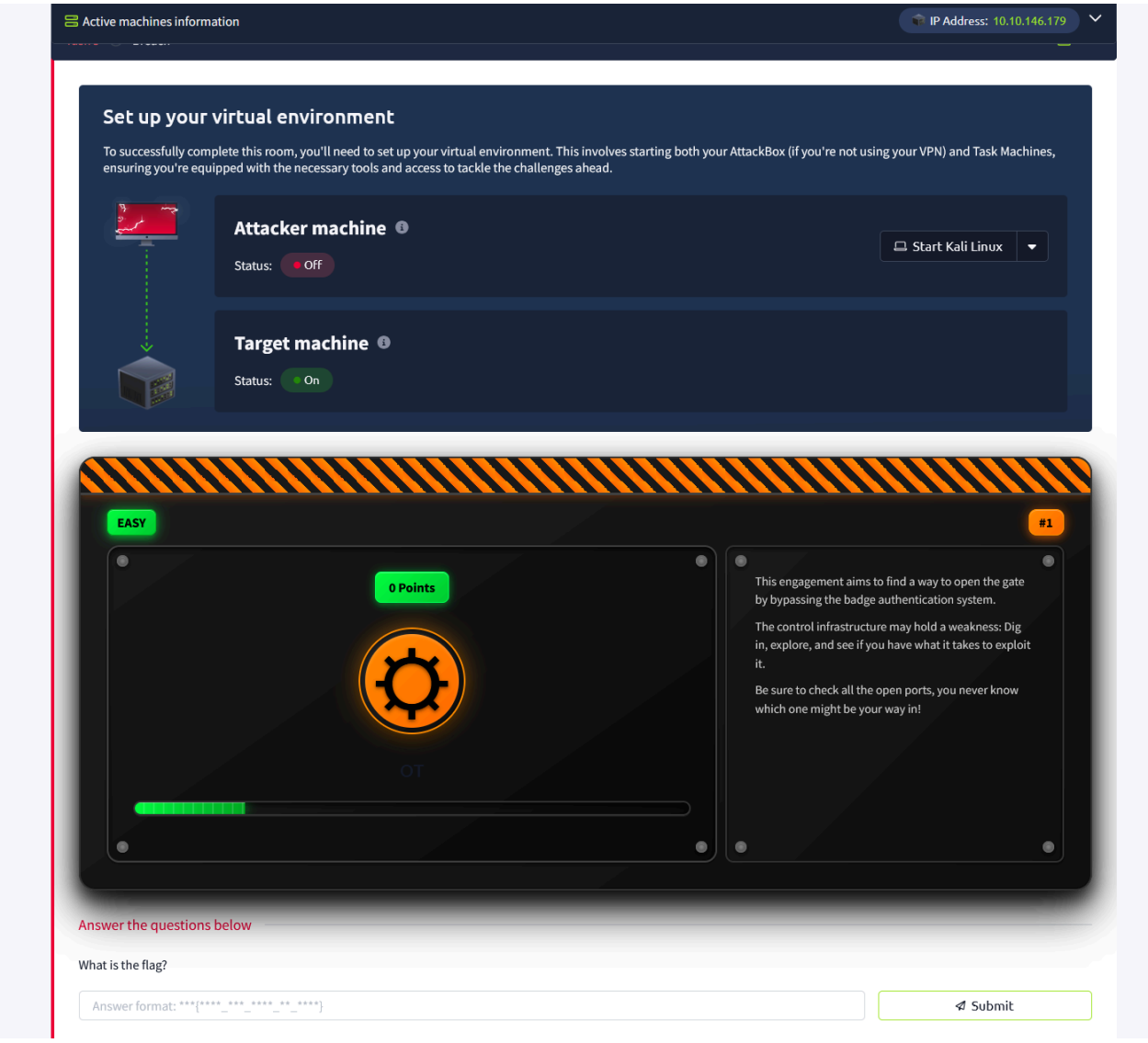
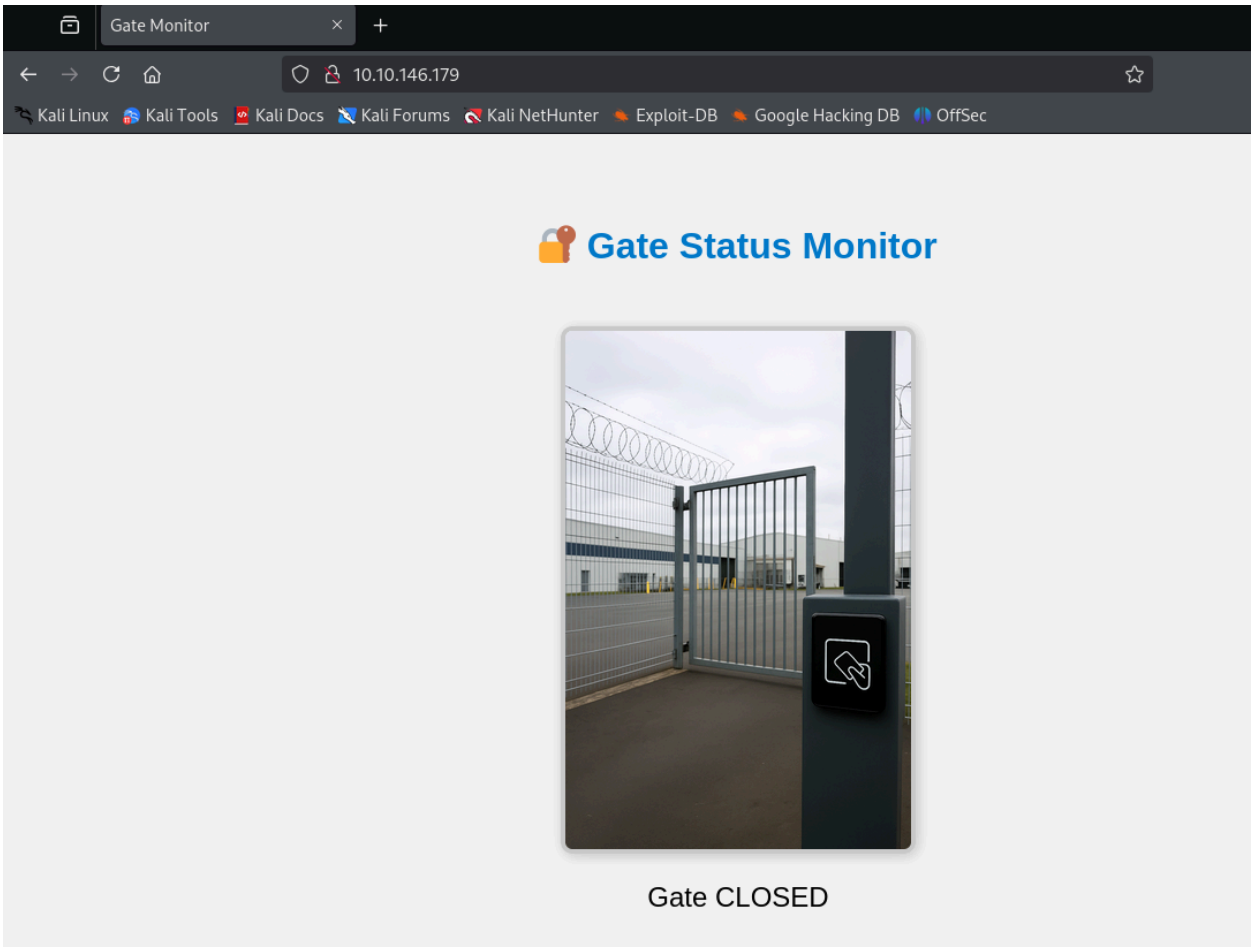
Industrial Intrusion CTF

Task 3 Breach

Getting introduced to the ctf, we are presented with our ip and hint. (Note that the ip may change throughout the writeup due to the machine expiring.)



Taking a look at the webpage, it looks like we are locked out of the system.

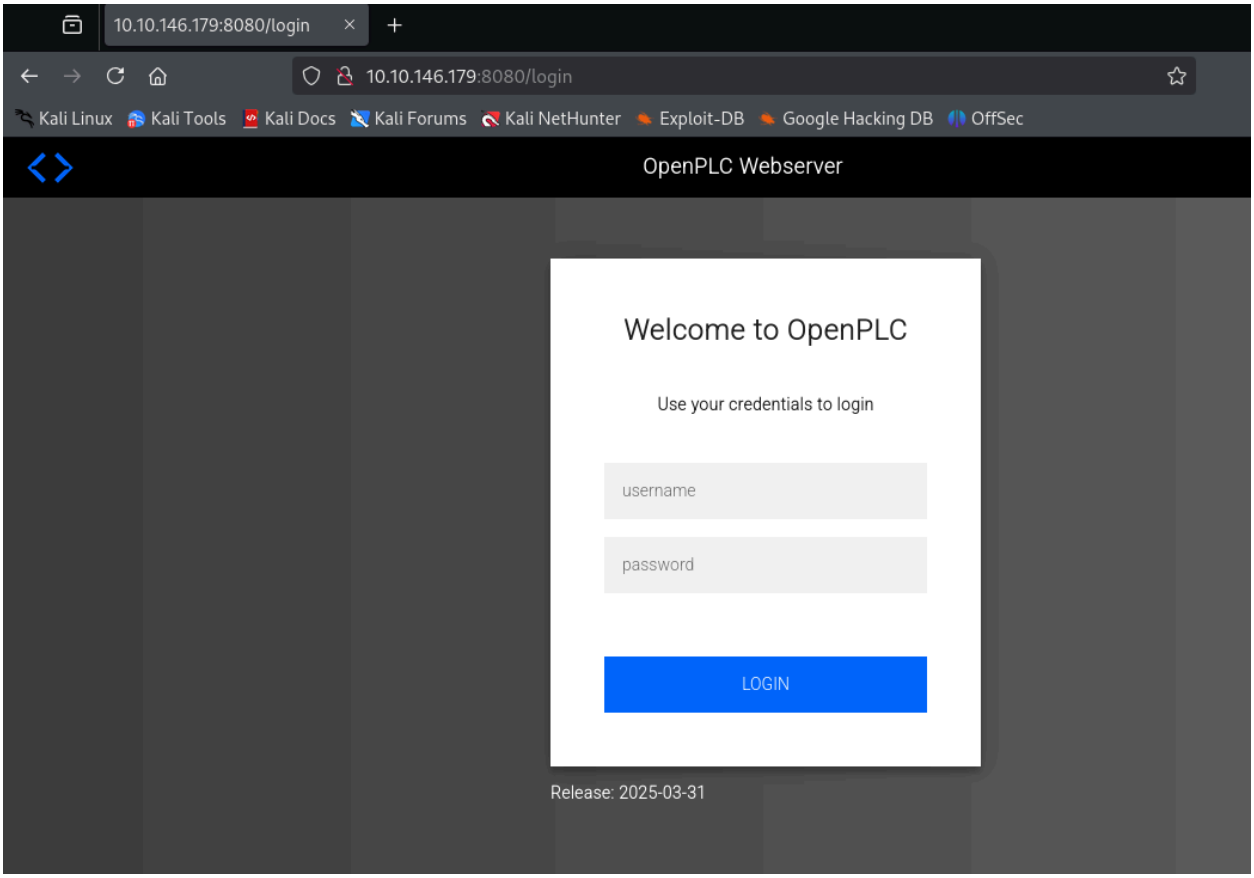


Running an nmap scan using `nmap 10.10.146.179 -sS -Pn -T5 -p-` to quickly see all open ports, we are presented with the following ports:

```
┌──(kali㉿kali)-[~/Desktop]
└─$ nmap 10.10.146.179 -sS -Pn -T5 -p-
Starting Nmap 7.95 ( https://nmap.org ) at 2025-06-25 20:50 EDT
Warning: 10.10.146.179 giving up on port because retransmission cap hit (2).
Nmap scan report for 10.10.146.179
Host is up (0.10s latency).
Not shown: 65528 closed tcp ports (reset)
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
102/tcp   open  iso-tsap
502/tcp   open  mbap
1880/tcp  open  vsat-control
8080/tcp  open  http-proxy
44818/tcp open  EtherNetIP-2

Nmap done: 1 IP address (1 host up) scanned in 104.63 seconds
```

We can see that port 80 gives is a closed gate, so I then check port 8080 to see what's there and we are presented with a login page.



I try to quickly login with ‘admin’ as username and password, but unfortunately it’s not that easy. I will keep this page open, but I do want to circle back and get a bit more information from the open ports.

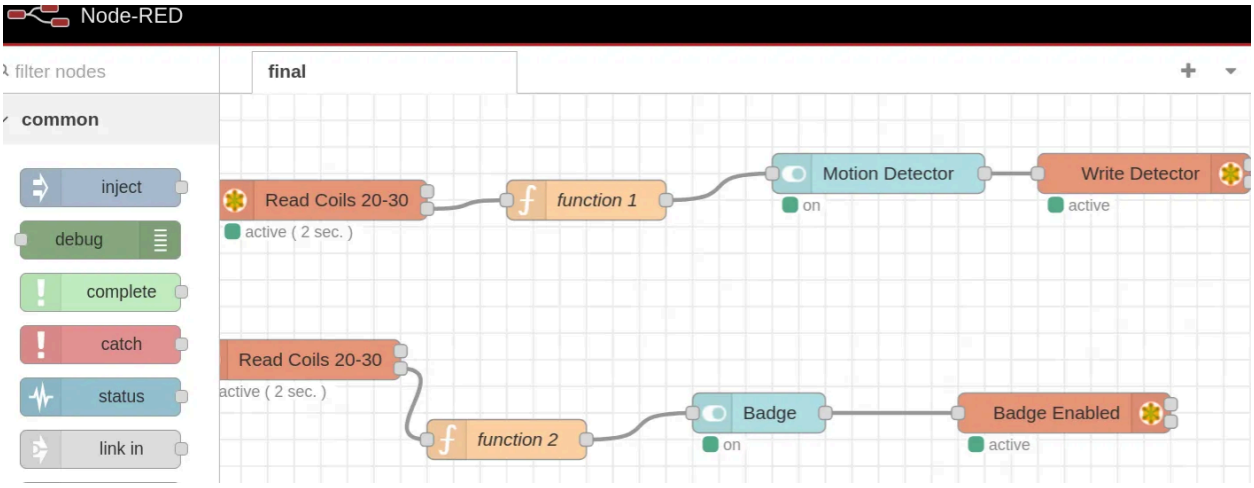
I then follow up with another nmap scan to gather more information

```
nmap 10.10.146.179 -sV -sC -p 22,80,102,502,1880,8080,44818
```

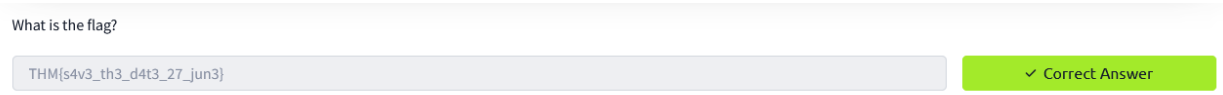
I can see that port `1880` has html contents in it

```
1880/tcp  open  vsat-control?
| fingerprint-strings:
|   DNSVersionBindReqTCP, RPCCheck:
|     HTTP/1.1 400 Bad Request
|     Connection: close
|   GetRequest:
|     HTTP/1.1 200 OK
|     Access-Control-Allow-Origin: *
|     Content-Type: text/html; charset=utf-8
|     Content-Length: 1733
|     ETag: W/"6c5-hGVEFL4qpfS9qVbAlfbm9AL7VT0"
|     Date: Thu, 26 Jun 2025 01:00:48 GMT
|     Connection: close
|     <!DOCTYPE html>
|     <html>
|     <head>
|     <meta charset="utf-8">
|     <meta http-equiv="X-UA-Compatible" content="IE=edge">
|     <meta name="viewport" content="width=device-width, initial-scale=1, max
imum-scale=1, user-scalable=0">
|     <meta name="apple-mobile-web-app-capable" content="yes">
|     <meta name="mobile-web-app-capable" content="yes">
|     <!--
|     Copyright OpenJS Foundation and other contributors, https://openjsf.org
/
|     Licensed under the Apache License, Version 2.0 (the "License");
|     this file except in compliance with the License.
|     obtain a copy of the License at
|     http://www.apache.org/licenses/LICENSE-2.0
|     Unless required by applicable law or agreed to in writing, softwa
|   HTTPOptions, RTSPRequest:
|     HTTP/1.1 204 No Content
|     Access-Control-Allow-Origin: *
|     Access-Control-Allow-Methods: GET,PUT,POST,DELETE
|     Vary: Access-Control-Request-Headers
|     Content-Length: 0
|     Date: Thu, 26 Jun 2025 01:00:48 GMT
|_    Connection: close
```

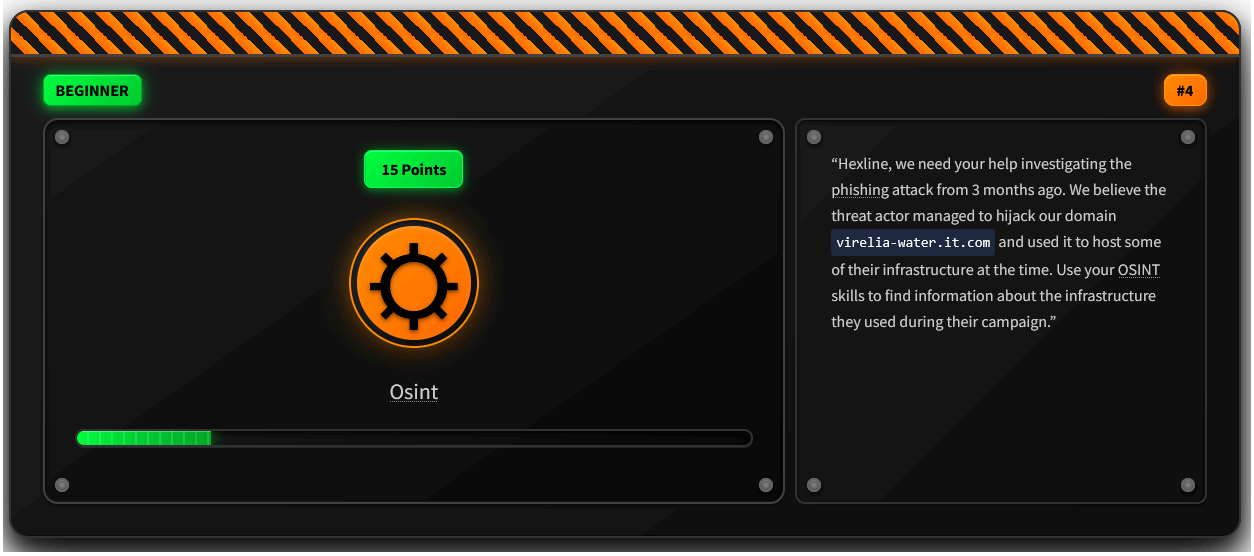
Port `1880` was hosting Node-RED, which showed us automation workflows.



At this point, I had to go to `http://10.10.146.179:1880/ui` where there was a UI switch to disable both motion detector and badge. This allowed me to go back to the gate status monitor and see that it was opened and the flag provided.



Task 5 OSINT 1



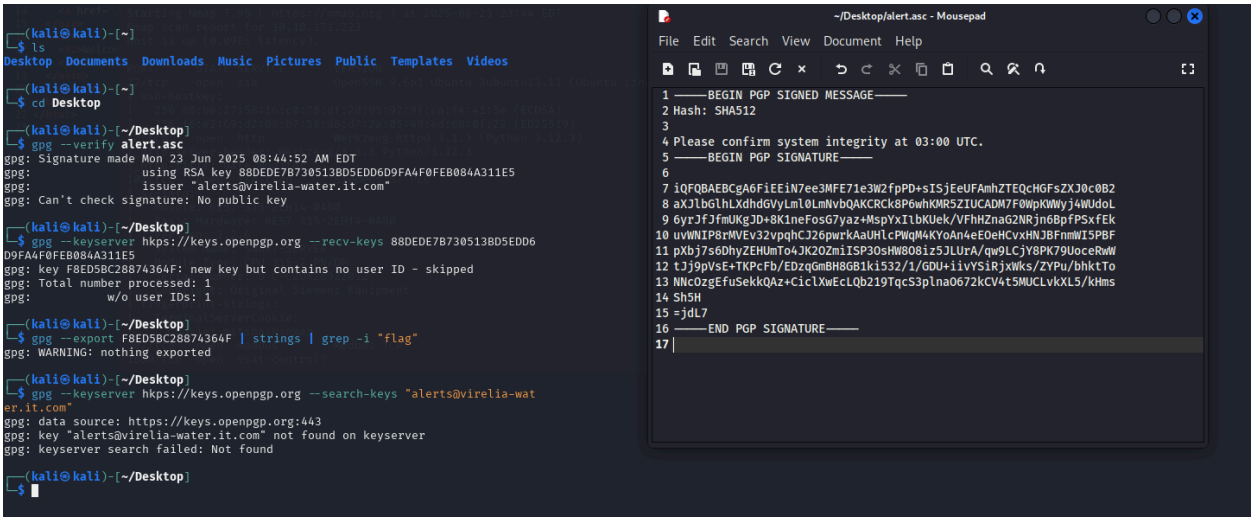
DNS lookup shows that it is linked to a github pages

Scanning the github, I found that there was a file that has been removed during an investigation according to push history

```
mail-archives/ot-alerts/2025-06.html
+ <html lang="en">
+ <head>
+   <meta charset="UTF-8">
+   <meta name="robots" content="index,follow">
+   <title>OT Alerts Exceptions – June 2025</title>
+   <link rel="stylesheet" href="/styles.css">
+ </head>
+ <body>
+   <header><h1>OT Alerts Exception Report – June 2025</h1></header>
+   <nav>
+     <a href="/">Home</a>
+     <a href="/mail-archives/">Archives Home</a>
+     <a href="/policies/">Compliance Policies</a>
+   </nav>
+   <main>
+     <p>This page lists <em>exceptional</em> OT-Alert messages for June 2025 only. Routine alerts have been redacted.</p>
+     <div class="message">
+       <div class="hdr">
+         From: DarkPulse &lt;alerts@virelia-water.it.com&gt;<br>
+         Date: Mon, 15 Jun 2025 02:15:00 +0000<br>
+         Subject: Scheduled OT Calibration
+       </div>
+       <pre>
+ -----BEGIN PGP SIGNED MESSAGE-----
+ Hash: SHA512
+
+ Please confirm system integrity at 03:00 UTC.
+ -----BEGIN PGP SIGNATURE-----
+
+ iQFQBAEBCgA6FiEEiN7ee3MFE71e3W2fpPD+sISjEeUFAmhZTEQcHGFsZXJ0c0B2
+ aXJlbGlhLXdhdGVyLml0LmNvbQAKCRCk8P6whKMR5ZIUCADM7F0WpKWwyj4WUdoL
+ 6yrJfJfmUKgJD+8K1neFosG7yaz+MspYxIlbKUek/VFhHZnaG2NRjn6BpfPSxfEk
+ uvWNIP8rMVEv32vpqhCJ26pwrkAaUHlcPWqM4KYoAn4eEOeHCvxHNJBFnmWI5PBF
+ pXbj7s6DhyZEHUmTo4JK2OZmiISP3OsHW8O8iz5JLUrA/qw9LCjY8PK79UoceRwW
+ tJj9pVsE+TKPcFb/EDzqGmBH8GB1ki532/1/GDU+iivYSiRjxWks/ZYPu/bhktTo
+ NNcOzgEfuSekkQAz+CiclXwEcLQb219TqcS3plnaO672kCV4t5MUCLvkXL5/kHms
+ Sh5H
+ =jdL7
+ -----END PGP SIGNATURE-----
+       </pre>
+     </div>
+   </main>
+   <footer>&copy; 2025 Virelia Water Control Facility</footer>
+ </body>
+ </html>
```

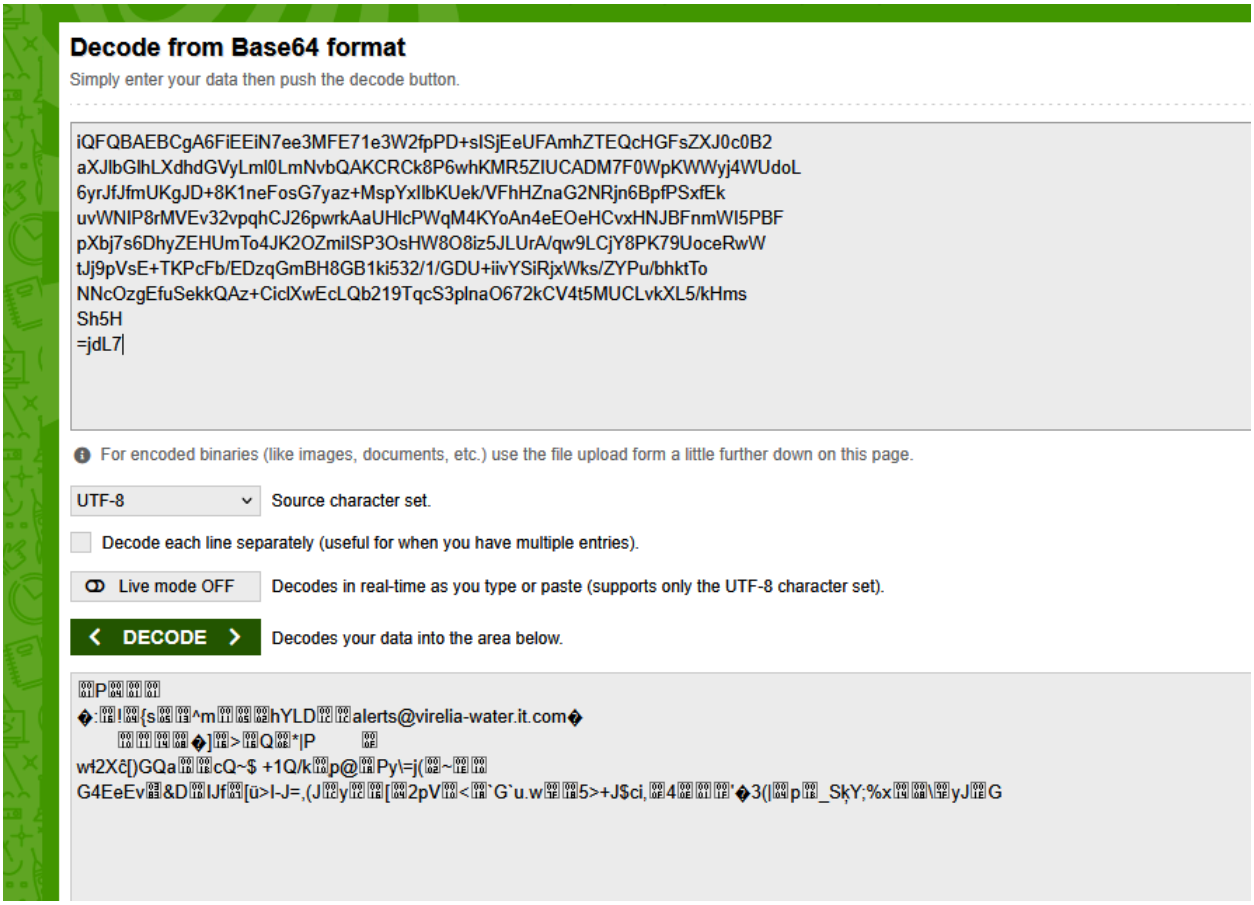
Since this was removed, I wanted to investigate the PGP Signature.

First I attempt to import the key to see it the flag may be somewhere in the key name.



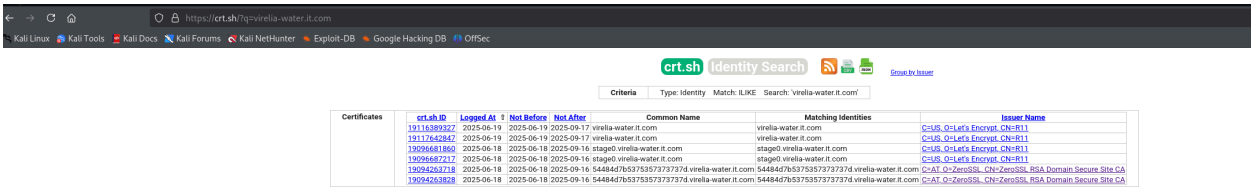
That led me to a dead end with a fake key and the email not giving me anything either.

Next, I take the signature and try to decrypt it using base64 decoding.

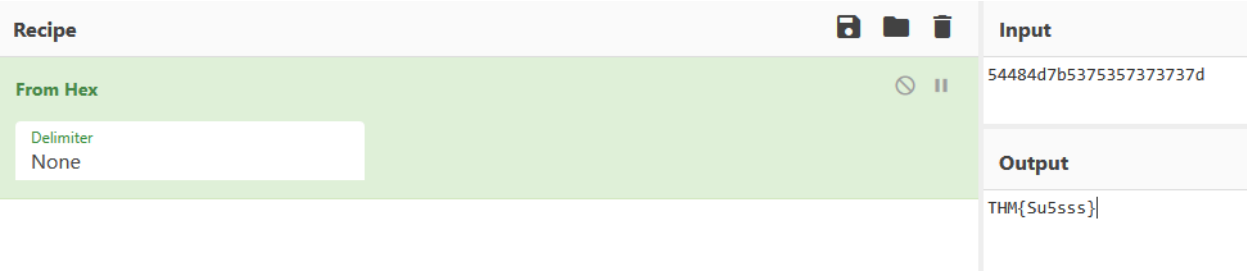


I'm surprised that an email has been decoded and it makes me think that a flag may be encoded like this, but I don't want to spend too long on what is very likely the wrong path.

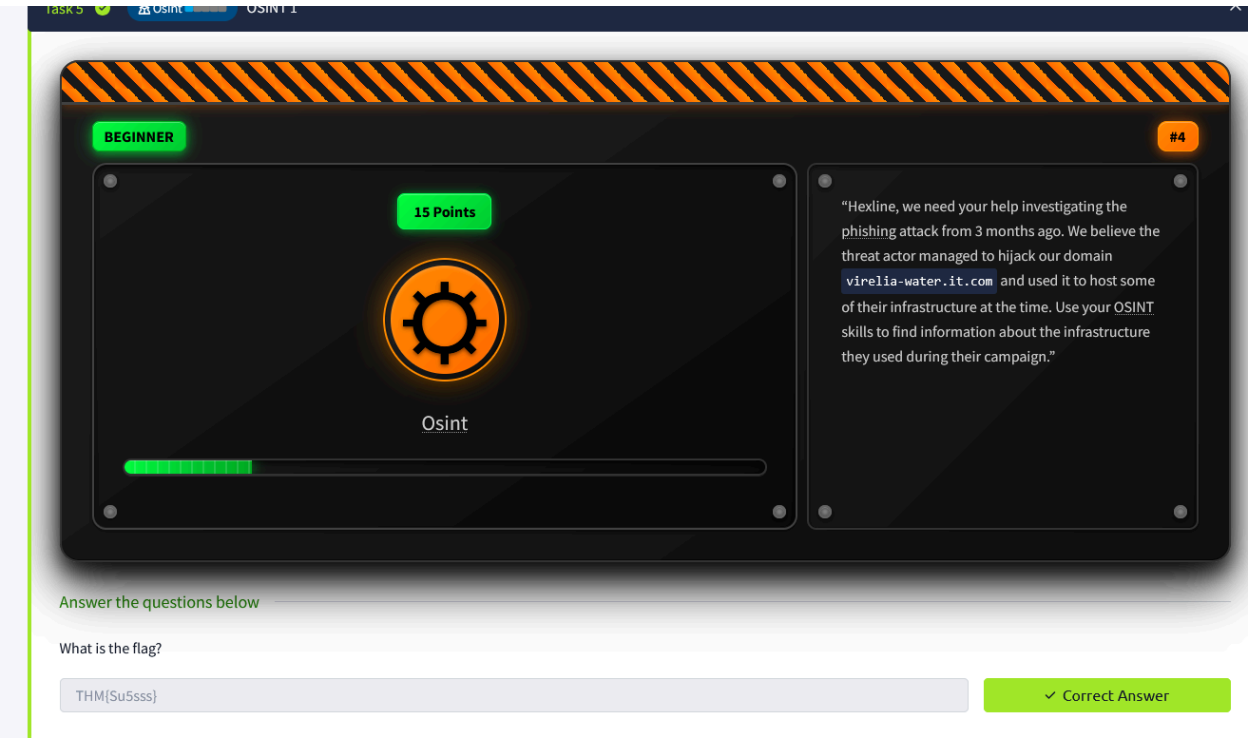
For an alternate approach, I wanted to check the website certificate to see all the subdomains for this website.



I came across `4484d7b5375357373737d.virelia-water.it.com` which seemed odd, so i decided to put the subdomain into cyberchef and see id it can decrypt it.



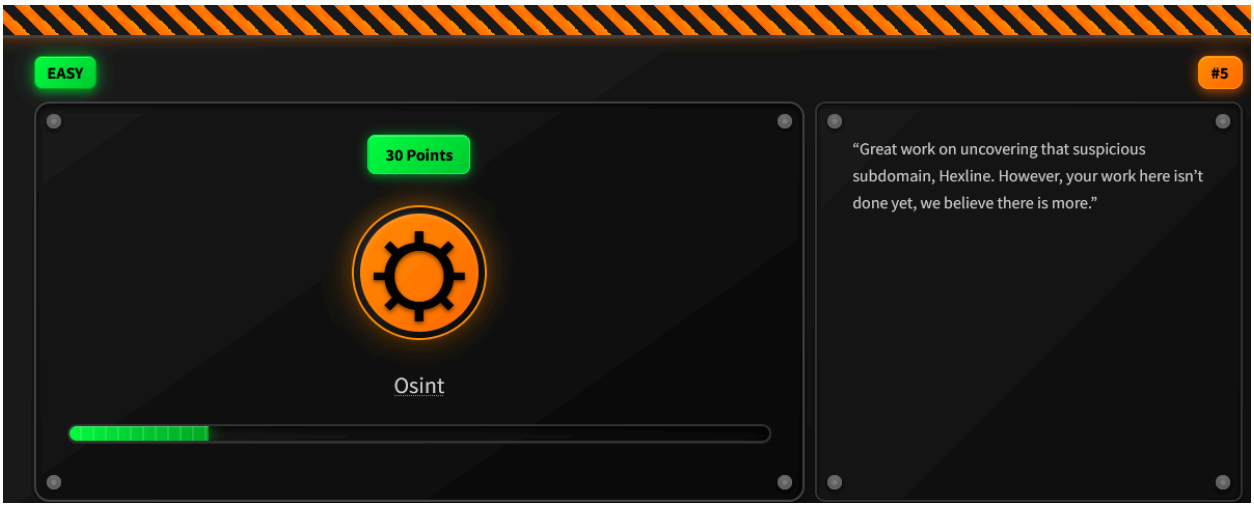
Sure enough, it looks like we have found our flag. Now it was time to check:



We have found the flag!

Task 6 OSINT 2

Our next task was to keep snooping around



I went back to the certificate subdomain list and saw `https://stage0.virelia-water.it.com/` which I decided to visit.

Looking at the source code, I saw that there was a link to a source script which looked interesting

```
    </div>
  </div>
  <script src="https://raw.githubusercontent.com/SanTzu/uplink-config/refs/heads/main/init.js"></script>
</body>
</html>
```

The script looked like:

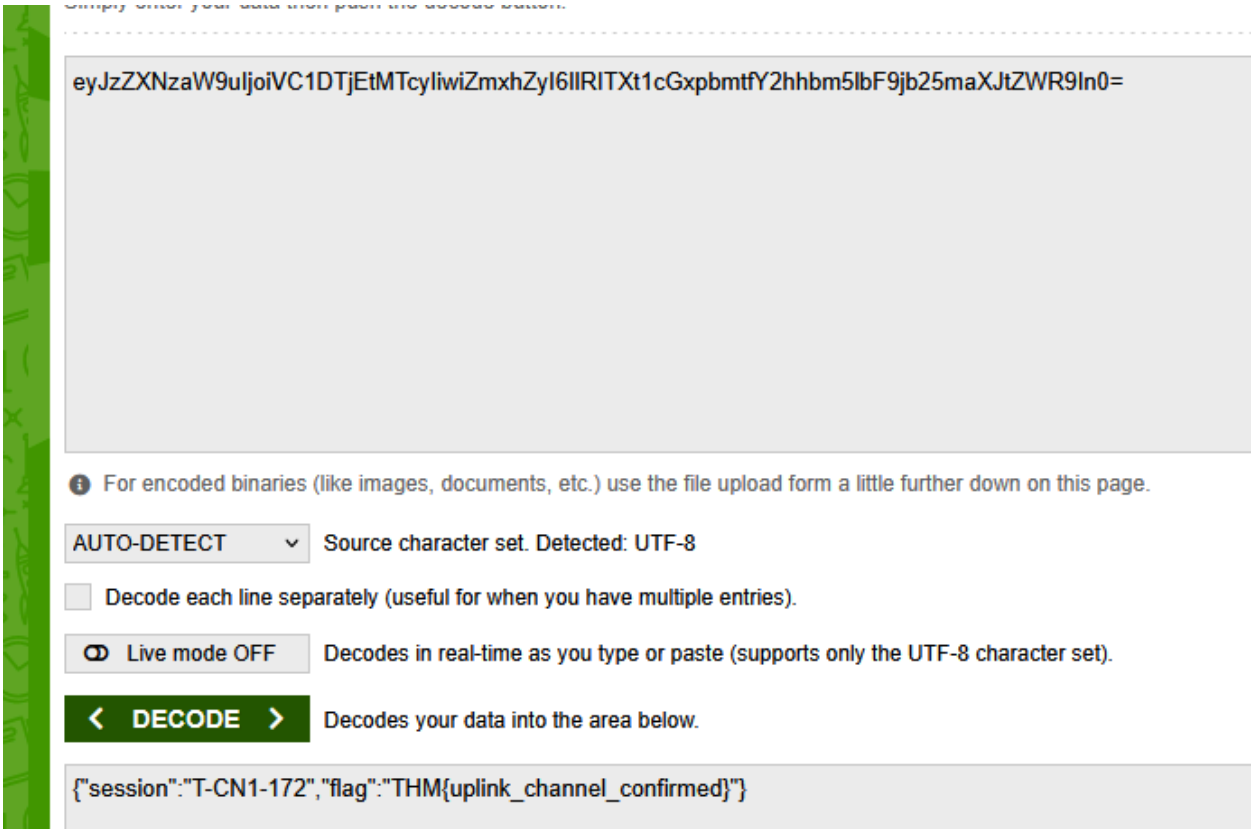
```
var beacon = {
  session_id: "O-TX-11-403",
  fallback_dns: "uplink-fallback.virelia-water.it.com",
  token: "JBSWY3DPEBLW64TMMQQQ==")
};
```

This was interesting because the initial error message on the website said that there was an error with the token. I then decided to do a dns lookup for the fallback_dns `uplink-fallback.virelia-water.it.com`

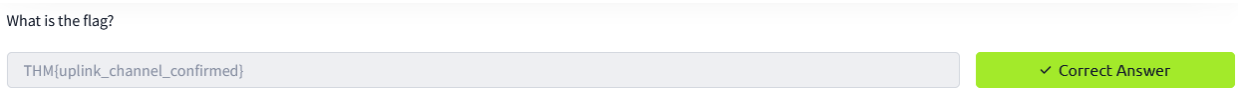
Domain Name: uplink-fallback.virelia-water.it.com
Dns Types: TXT=16

Name: uplink-fallback.virelia-water.it.com
Type: 16
TTL: 1799
rRset Type: 16
Strings: eyJzZXNzaW9uIjoiVC1DTjEtMTcyIiwiZmxhZyI6IlRITXt1cGxpbmtfY2hhbm5lbF9jb25maXJtZWR9In0=
Raw Text: uplink-fallback.virelia-water.it.com. 1799 IN TXT
"eyJzZXNzaW9uIjoiVC1DTjEtMTcyIiwiZmxhZyI6IlRITXt1cGxpbmtfY2hhbm5lbF9jb25maXJtZWR9In0="

This led me to see that there was something which seemed to be base64 encrypted, so I decrypted the string.



It looks like I have found a flag! Now I just confirmed that it would be for this task.



Task 7 OSINT 3

This task seems to be related to the path I was going down with the first OSINT task and the PGP signing.

I then started my investigation at the github commit with the PGP signing
https://github.com/virelia-water/compliance/commit/6d355c02e0e08525712fbd720695acd0450a067a

1 file changed +46 -0 lines changed

mail-archives/ot-alerts/2025-06.html

```
+ </head>
+ <body>
+   <header><h1>OT Alerts Exception Report – June 2025</h1></header>
+   <nav>
+     <a href="/">Home</a>
+     <a href="/mail-archives/">Archives Home</a>
+     <a href="/policies/">Compliance Policies</a>
+   </nav>
+   <main>
+     <p>This page lists <em>exceptional</em> OT-Alert messages for June 2025 only. Routine alerts have been redacted.</p>
+     <div class="message">
+       <div class="hdr">
+         From: DarkPulse &lt;alerts@virelia-water.it.com&gt;<br>
+         Date: Mon, 15 Jun 2025 02:15:00 +0000<br>
+         Subject: Scheduled OT Calibration
+       </div>
+       <pre>
+ -----BEGIN PGP SIGNED MESSAGE-----
+ Hash: SHA512
+
+ Please confirm system integrity at 03:00 UTC.
+ -----BEGIN PGP SIGNATURE-----
+
+ iQFQBAEBCgA6FiEEiN7ee3MFE71e3W2fpPD+sISjEeUFAmhZTEQcHGFsZXJ0c0B2
+ aXJlbGlhLXdhdGVyLml0LmNvbQAKCRCk8P6whKMR5ZIUCADM7F0WpKWWyj4WUdoL
+ 6yrJfJfmUKgJD+8K1neFosG7yaz+MspYxIlbKUek/VFhHZnaG2NRjn6BpfPSxfEk
+ uvWNIP8rMVEv32vpqhCJ26pwrkAaUHlcPWqM4KYoAn4eEOeHCvxHNJBFnmWI5PBF
+ pXbj7s6DhyZEHUmTo4JK2OZmiISP3OsHW8O8iz5JLUrA/qw9LCjY8PK79UoceRwW
+ tJj9pVsE+TKPcFb/EDzqGmBH8GB1ki532/1/GDU+iivYSiRjxWks/ZYPu/bhktTo
+ NNcOzgEfuSekkQAz+CiclXwEcLQb219TqcS3plnaO672kCV4t5MUCLvkXL5/kHms
+ Sh5H
+ =jdL7
+ -----END PGP SIGNATURE-----
+       </pre>
+     </div>
+   </main>
+   <footer>&copy; 2025 Virelia Water Control Facility</footer>
+ </body>
+ </html>
```

Now having that open, I would like to check if any of the keyservers have any keys related to DarkPulse

```
┌──(kali㉿kali)-[~/Downloads]
└─$ gpg --keyserver keyserver.ubuntu.com --search-keys "DarkPulse"
gpg: data source: http://185.125.188.26:11371
(1)     Ghost (THM{h0pe_th1s_k3y_doesnt_le4d_t0_m3}) <solstice.tech.ops@gmail
.
        DarkPulse (THM{h0pe_th1s_k3y_doesnt_le4d_t0_m3}) <alerts@virelia-wate
r
          2048 bit RSA key F8ED5BC28874364F, created: 2025-06-23
Enter number(s), N)ext, or Q)uit >
```

Well, it looks like I have found the flag I was looking for.

What is the flag?

THM{h0pe_th1s_k3y_doesnt_le4d_t0_m3} ✓ Correct Answer

Task 24 Reverse Engineering “Auth”

This task requires us to reverse engineer an authentication

With the zip file downloaded, I unzipped it and decompiled it using ghidra. There we can see that the main gist of the program is that it asks for an unlock code and will output the flag to me if it is the correct input.

```
int iVar1;
char *pcVar2;
undefined8 uVar3;
size_t sVar4;
FILE *__stream;
long in_FS_OFFSET;
undefined8 local_168;
undefined8 local_160;
undefined8 local_158 [8];
char flag [264];
long local_10;

local_10 = *(long *)(in_FS_OFFSET + 0x28);
local_160 = 0xefcdab8967452301;
printf("[?] Enter unlock code: ");
pcVar2 = fgets((char *)local_158,0x40,stdin);
if (pcVar2 == (char *)0x0) {
  fwrite("Error reading input\n",1,0x14,stderr);
  uVar3 = 1;
}
else {
  sVar4 = strcspn((char *)local_158,"\r\n");
  *(undefined1 *)((long)local_158 + sVar4) = 0;
  sVar4 = strnlen((char *)local_158,0x40);
  if (sVar4 == 8) {
    local_168 = local_158[0];
    transform(&local_168,8);
    iVar1 = memcmp(&local_168,&local_160,8);
    if (iVar1 == 0) {
      __stream = fopen("flag.txt","r");
      if (__stream == (FILE *)0x0) {
        perror("fopen");
        uVar3 = 1;
      }
      else {
        pcVar2 = fgets(flag,0x100,__stream);
        if (pcVar2 == (char *)0x0) {
          fwrite("Error reading flag\n",1,0x13,stderr);
        }
        else {
          printf("[+] Access Granted! Flag: %s",flag);
        }
        fclose(__stream);
        uVar3 = 0;
      }
    }
    else {
      puts("[!] Access Denied!");
      uVar3 = 1;
    }
  }
```

There is a `transform()` function as well which seems to XOR values with the hex value of 0x55

```
void transform(long param_1,ulong param_2)

{
  undefined8 local_10;

  for (local_10 = 0; local_10 < param_2; local_10 = local_10 + 1) {
    *(byte *)(local_10 + param_1) = *(byte *)(local_10 + param_1) ^ 0x55;
  }
  return;
}
```

Looking at the main function closely now, we can see that our input should be 8 characters long. Also, we see that there is `local_160 = 0xefcdab8967452301;` set at the top of our program, which happens to also be 8 characters. My understanding is that this variable gets put through the transform function so that each character is XORed by 0x55, and then that is compared to our input. Therefore, we have to perform the transformation:

0x01 XOR 0x55 = 0x54

0x23 XOR 0x55 = 0x76

0x45 XOR 0x55 = 0x10

0x67 XOR 0x55 = 0x32

0x89 XOR 0x55 = 0xdc

0xab XOR 0x55 = 0xfe

0xcd XOR 0x55 = 0x98

0xef XOR 0x55 = 0xba

To input the code into the program, I would need to use the following code

```
python -c "import sys; sys.stdout.buffer.write(b'\x54\x76\x10\x32\xdc\xfe\x98
```

```
\xba\n')" | nc 10.10.253.80 9005
```

```
┌──(kali㉿kali)-[~/Desktop]
└─$ python -c "import sys; sys.stdout.buffer.write(b'\x54\x76\x10\x32\xdc\xfe
\x98\xba\n')" | nc 10.10.253.80 9005
[?] Enter unlock code: [+] Access Granted! Flag: THM{Simple_tostart_nice_done
_mwww}
```

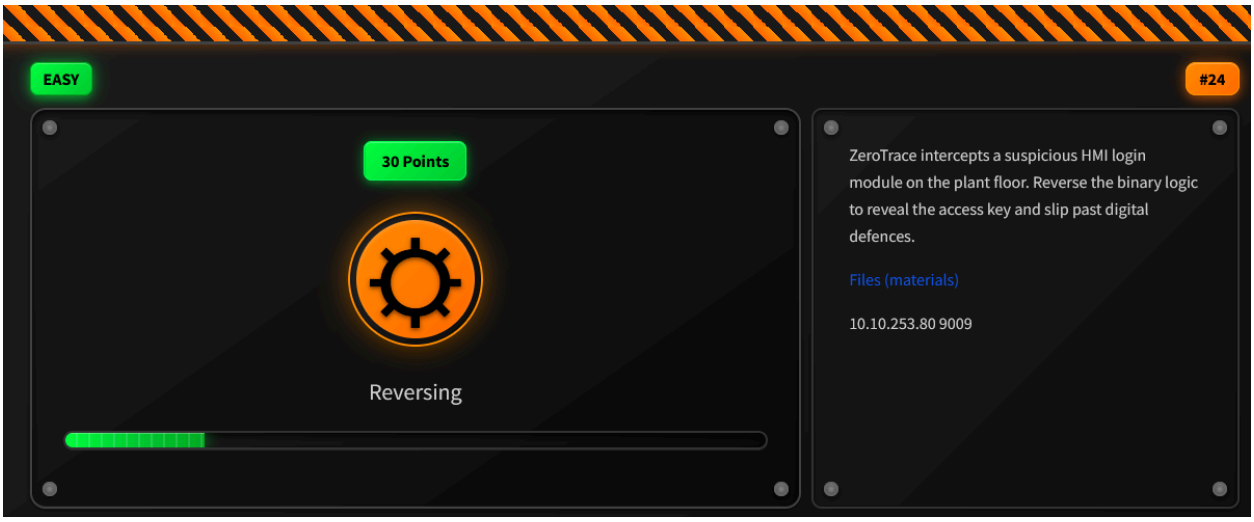
This seems to have been successful and the flag was provided.

What is the content of flag.txt?

THM{Simple_tostart_nice_done_mwww}

✓ Correct Answer

Task 25 Reverse Engineering "Access Granted"



This task provides files, which I unpack and decompile with ghidra.

```c
undefined8 main(void)

{
  int iVar1;
  long in_FS_OFFSET;
  char local_38 [40];
  long local_10;

  local_10 = *(long *)(in_FS_OFFSET + 0x28);
  setvbuf(stdout,(char *)0x0,2,0);
  setvbuf(stdin,(char *)0x0,2,0);
  printf("Enter the password : ");
  read(0,local_38,0x1f);
  printf("\nprocessing...");
  iVar1 = strncmp(pass,local_38,10);
  if (iVar1 == 0) {
    puts("Access Granted!");
    print_flag();
  }
  else {
    puts("\nWrong Password!");
  }
  if (local_10 != *(long *)(in_FS_OFFSET + 0x28)) {
                    /* WARNING: Subroutine does not return */
    __stack_chk_fail();
  }
  return 0;
}
```

What I initially see is that there is a string compare of 10 characters between out input and a `pass` variable. So I check what the variable contains.

```
                    pass                                        XREF[2]:     Entry Point(*), main:001013b1(*)
00104010 69 6e 64        ds          "industrial"
         75 73 74
         72 69 61 ...
```

It seems like the string “industrial” is stored there, which is a 10 character string. So if I start my password with “industrial” it should be successful. I then try it out:

```
┌──(kali㉿kali)-[~/Desktop]
└─$ echo "industrial" | nc 10.10.194.199 9009
Enter the password :
processing ... Access Granted!
THM{s0meth1ng_inthe_str1ng_she_knows}

┌──(kali㉿kali)-[~/Desktop]
└─$ echo "industrialrevolution" | nc 10.10.194.199 9009
Enter the password :
processing ... Access Granted!
THM{s0meth1ng_inthe_str1ng_she_knows}
```

It seems to work and gives me the flag.

What is the content of the file flag.txt?

THM{s0meth1ng_inthe_str1ng_she_knows} ✓ Correct Answer

Final result

With the above flags, I was able to rank #521

Rank	Username	[Task 1] #1	[Task 2] #1	[Task 3] #1	[Task 4] #1	[Task 5] #1	[Task 6] #1	[Task 7] #1
521	g00fyg00bers	✓	✓	✓	✓	✓	✓	✓