

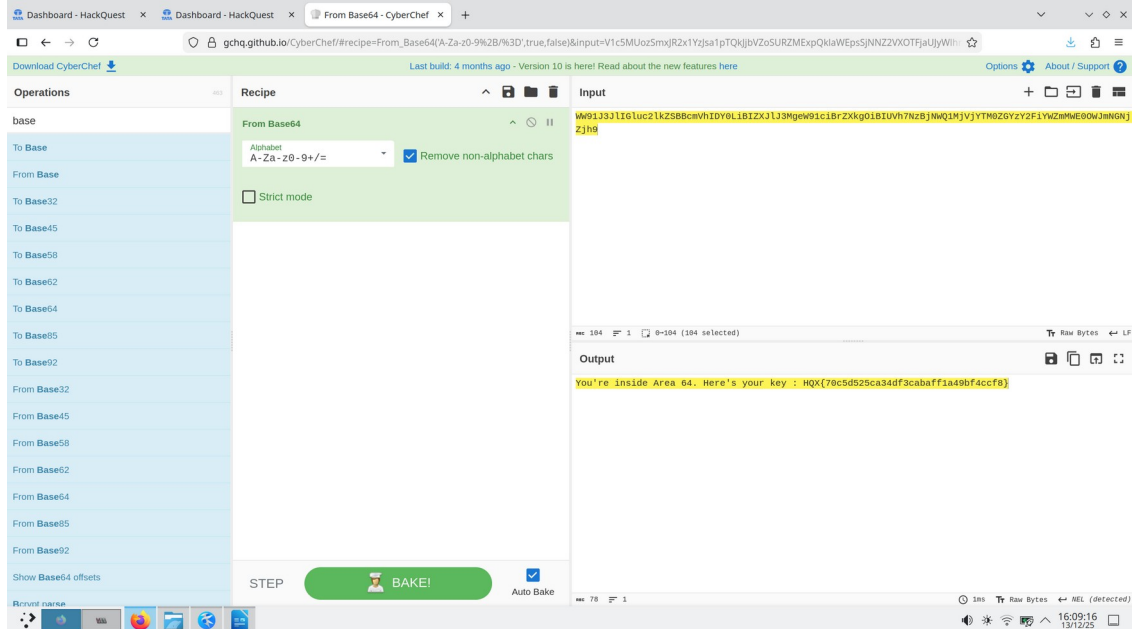
# Round 1 report

# tcs HackQuest

## Season 10

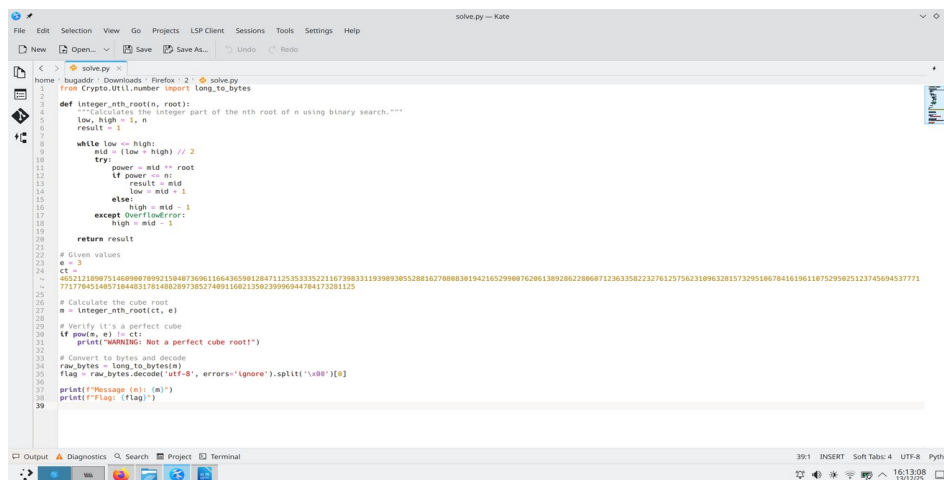
**Contest Date: - 13th December 2025**

CT ID	REDACTED
Name	Pranay Pawar
College/University	REDACTED
City	REDACTED
Challenges solved & the total score	4 (700)
Anything else that you want us to know	

**Challenge Title: Area 64****Flag: HQX{70c5d525ca34df3cabaff1a49bf4ccf8}****Approach (Step by Step):****1. Decode the Base64**

**Challenge Title: Small-E****Flag: HQX{36b8682966c5dabfbac72d3f687a77ff}****Approach (Step by Step):**

1. Made the script for RSA low exponent attack: with small public exponent ( $e=3$ ) and no padding, ciphertext equals plaintext<sup>3</sup>, so attacker recovers the message by taking the integer cube root
2. [Code: <https://termbin.com/af21> ]

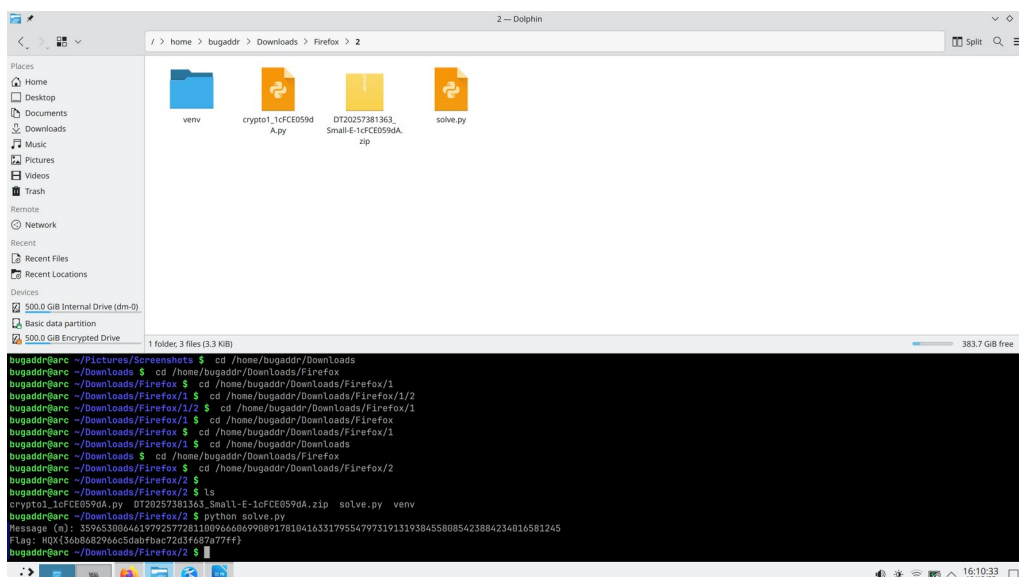


```

1  # solve.py
2  from Crypto.Util.number import long_to_bytes
3
4  def integer_nth_root(n, root):
5      """Calculates the integer part of the nth root of n using binary search."""
6      low, high = 1, n
7      result = 1
8
9      while low <= high:
10         mid = (low + high) // 2
11         try:
12             power = mid ** root
13             if power == n:
14                 result = mid
15                 low = mid + 1
16             else:
17                 high = mid - 1
18             except OverflowError:
19                 high = mid - 1
20
21     return result
22
23 # Given values
24 e = 3
25 ct = 405212808751460808789921504073096116643650812047112535335221167380331193989385528816278088301942165299607628613892862286687123633582232761257562310963281573295106784161961107529582512374560453771
77177805480710445373148639738527489116821506239969944784172831125
26
27 # Calculate the cube root
28 n = integer_nth_root(ct, e)
29
30 # Verify it's a perfect cube
31 if pow(n, e) != ct:
32     print("WARNING: Not a perfect cube root!")
33
34 # Convert to bytes and decode
35 raw_bytes = long_to_bytes(n)
36 flag = raw_bytes.decode('utf-8', errors='ignore').split('\x00')[0]
37
38 print("Message (n):")
39 print(flag)

```

3. Run the script and got flag



```

bugaddr@bugaddr:~/Downloads/Firefox/2 $ cd /home/bugaddr/Downloads/Firefox/2
bugaddr@bugaddr:~/Downloads/Firefox/2 $ python solve.py
Message (n): 3526530846479257722118946606990891781041633179547973191319384558085423884234016581245
Flag: HQX{36b8682966c5dabfbac72d3f687a77ff}
bugaddr@bugaddr:~/Downloads/Firefox/2 $

```

**Challenge Title: Hidden Layers****Flag: HQX{24c0ce09e08bcb246d09c1439d1d48f0}****Approach (Step by Step):**

1. **LSB steganography:** hidden data extracted from the least significant bits of image pixels using zsteg, revealing an embedded text/flag without altering visible image quality.

```

kali@kali: ~/Downloads
$ find -name zsteg
kali@kali: ~/Downloads
$ ls
dd.png
kali@kali: ~/Downloads
$ ././local/share/gem/ruby/3.3.0/bin/zsteg dd.png
b1,rgb,lsb,xy .. text: HQX{24c0ce09e08bcb246d09c1439d1d48f0}
b2,r,msb,xy .. text: ["U" repeated 14 times]
b2,g,lsb,xy .. text: ["U" repeated 22 times]
b2,g,msb,xy .. text: ["U" repeated 14 times]
b2,b,lsb,xy .. text: ["U" repeated 26 times]
b2,b,msb,xy .. text: ["U" repeated 12 times]
b3,g,lsb,xy .. file: very old 16-bit-int big-endian archive
b4,r,lsb,xy .. text: "DL0L000L00DL0"
b4,r,msb,xy .. text: "DL0L000L00DL0"
b4,g,lsb,xy .. text: "f" repeated 28 times
b4,g,msb,xy .. text: "f" repeated 28 times
b4,b,lsb,xy .. text: "U" repeated 28 times
b4,b,msb,xy .. text: "U" repeated 28 times
kali@kali: ~/Downloads

```

**Challenge Title: Seeds of Time****Flag: HQX{c126bb454bf27489a1583af0729fbd08}****Approach (Step by Step):**

1. Built this script for **time-based PRNG seed brute-force attack** and tried to attack last 30days PRNG [Code: <https://termbin.com/zqza> ]

```

1  solve.py
2  import random
3  import time
4  from datetime import datetime
5
6  # == 1. REPLACE THIS WITH YOUR CTF OUTPUT ==
7  # =====
8
9  # 1. The hex-encoded ciphertext from the challenge output
10 CIPHER_HEX = "HQBx{c126bb454bf27489a1583af0729fbd08}3062366b8d1c"
11
12 # 2. Approximate time of generation (keep this as the time you are running the script)
13 APPROX_SEED = int(time.time())
14
15 # =====
16
17 # == INCREASED WINDOW SIZE ==
18 # time approximately 30 days (2,592,000 seconds) of past seeds.
19 WINDOW_SIZE = 2592000
20
21 # -- Decryption Logic --
22 try:
23     cipher = bytes.fromhex(CIPHER_HEX)
24 except ValueError:
25     print("ERROR: Please replace CIPHER_HEX with a valid hexadecimal string.")
26     exit()
27
28 cipher_len = len(cipher)
29
30 def generate_keystream(length, seed):
31     """Generate the keystream using the given seed, mimicking the CTF script."""
32     random.seed(seed)
33     return bytes([random.randrange(256) for _ in range(length)])
34
35 def decrypt(cipher, keystream):
36     """Perform the XOR operation: Plaintext = Ciphertext XOR Keystream."""
37     return bytes([c ^ k for c, k in zip(cipher, keystream)])
38
39 print("[*] Starting attack on a {} second window around {}".format(WINDOW_SIZE, datetime.fromtimestamp(APPROX_SEED)))
40
41 # Brute-force the possible integer seeds in the time window.
42 # Note: Since we know the current time (least known) to the past.
43 start_time = APPROX_SEED - WINDOW_SIZE // 2
44 end_time = APPROX_SEED + WINDOW_SIZE // 2
45
46 for current_seed in range(start_time, end_time, 1):
47     keystream = generate_keystream(cipher_len, current_seed)
48     plaintext = decrypt(cipher, keystream)
49
50 # Check if the decrypted byte line is a readable flag format.
51 if plaintext.startswith(b"QBX{") or plaintext.startswith(b"FLAG{") or all(32 < b < 128 for b in plaintext if b not in (0, 16, 33)):
52     print("[*] FLAG FOUND [{}]" .format(current_seed))
53     print("[*] Decrypted flag: {}".format(plaintext.decode()))
54     print("[*] Recovered Seed: {}".format(current_seed))
55     break
56
57 else:
58     print("[*] Attack failed. The challenge creation date is outside the 30-day window. Try increasing WINDOW_SIZE further.")

```

2. The flag

```

bugadd@bugadd:~/Pictures$ cd /home/bugadd/Pictures/Screenshots
bugadd@bugadd:~/Pictures/Screenshots$ cd /home/bugadd/Downloads
bugadd@bugadd:~/Downloads$ cd /home/bugadd/Downloads/Firefox
bugadd@bugadd:~/Downloads/Firefox$ cd /home/bugadd/Downloads/Firefox/5
bugadd@bugadd:~/Downloads/Firefox/5$ cd /home/bugadd/Downloads/Firefox/5
bugadd@bugadd:~/Downloads/Firefox/5$ cd /home/bugadd/Downloads/Firefox/5
bugadd@bugadd:~/Downloads/Firefox/5$ python solve.py
[*] Starting attack on a 2592000 second window around 2025-12-13 16:19:57...
[***] FLAG FOUND [***]
[*] Recovered Seed: 1765800209 (2025-12-13 18:00:09)
[*] Decrypted flag: HQX{c126bb454bf27489a1583af0729fbd08}
bugadd@bugadd:~/Downloads/Firefox/5$ cd /home/bugadd/Pictures
bugadd@bugadd:~/Pictures$ cd /home/bugadd/Pictures/Screenshots
bugadd@bugadd:~/Pictures/Screenshots$ cd /home/bugadd/Pictures
bugadd@bugadd:~/Pictures$ cd /home/bugadd/Pictures
bugadd@bugadd:~/Pictures$ cd /home/bugadd/Downloads/Firefox/5
bugadd@bugadd:~/Downloads/Firefox/5$

```