Final CTF

~Purva Naresh Rumde (pr23b)

Problem ID	Captured Flag/Answer	Steps
WEB BeepBoop	fsuCTF{th3_r0b0t5_4r3_t4k1ng_0v3r}	In this problem, I obtained the access to a webpage using robots.txt and got final.html which contained the flag.
WEB The path less Traversed	fsuCTF{r34d1ng_r41nb0w}	In this problem the webpage consisted of a input box. And a few files were mentioned along with it. Here I used the directory method and found my way back to the flags.txt with the help of the input box provided there.
CRYPTO shifty	fsuCTF{1t5_4ll_4b0ut_3nd1an3sses}	In this problem I unscrambled the flag by reversing every three letters.
CRYPTO Skill issue	fsuCTF{x0r_sk1ll_ch3ck}	In this problem there was a ciphertext that was provided. After analyzing it I figured out it a base64 encoding. Using Cyberchef I decoded the string. Again applied XOR bruteforce on the decoded string and that is how I got the flag.
CRYPTO Really secure algorithm	fsuCTF{rsa_fun}	In this problem the txt consisted of the values of a RSA cipher. I simply put those values on the site. It didn't work so I changed the value of E to it default value that is always there for every problem and that is how I got the flag.
PWN menu	fsuCTF{N0t_4_L0T_0f_c0v3R493}	I wrote a python script for this problem This Python script connects to a server at "ctf.cs.fsu.edu" on port 32333. It sends a string of 100 'A's to the server, followed by a newline character. Then it shuts down the write side of the socket and waits for a response from the server. Finally, it prints the response and closes the connection

FORENSICS Time Capsule	fsuCTF{t!m3_c4psule5_c4n_h0ld_m4ny_s3cr3ts}	In this problem I used Kali. First I unzipped the file that was provided it was an image. After unzipping there were a few files they were to disguise. The main was audio file. I put this audio file in Audacity. After applying spectrogram the waves produced the flag.
REVERSE ENG Mason-Ball	fsuCTF{citric_acid_is_used_for_canning_I_think}	In this problem the jar file was provided I simply extracted the file. It consisted of a class file. I ran the program and that is how I got the flag.
REVERSE ENG Rev your engines	fsuCTF{a1w4ys_ch3ck_str!ngs_f1r$t}	In this problem I ran the strings command on this file and got a list of strings. After analyzing the strings. A group of them seemed like a base64 encoded string. So I simply decoded them on Cyberchef and that is how I got the flag.
MISC Who is the coolest	flag{dawson}	He is the coolest
MISC Who is your favorite member	fsuCTF{dawson}	Since he is the coolest, he’s also my favourite

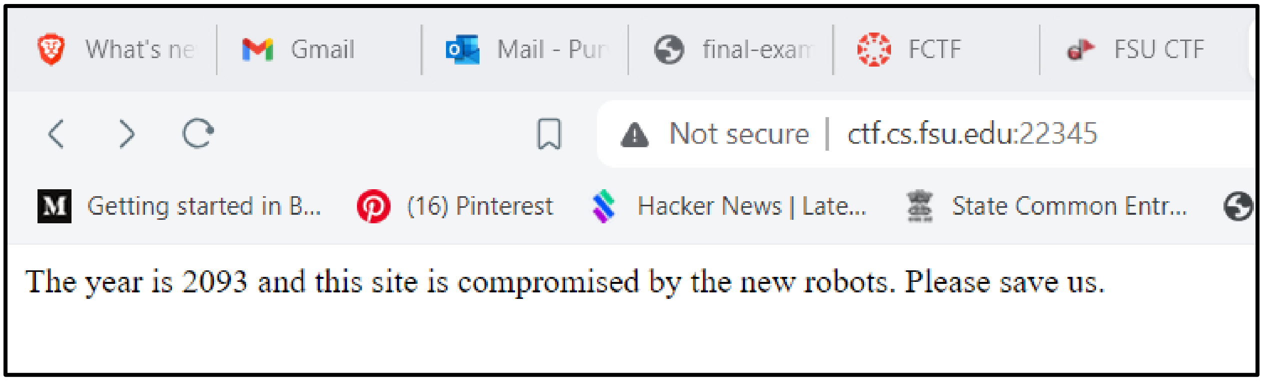
WEB

1. **Beep boop: fsuCTF{th3_r0b0t5_4r3_t4k1ng_0v3r}**

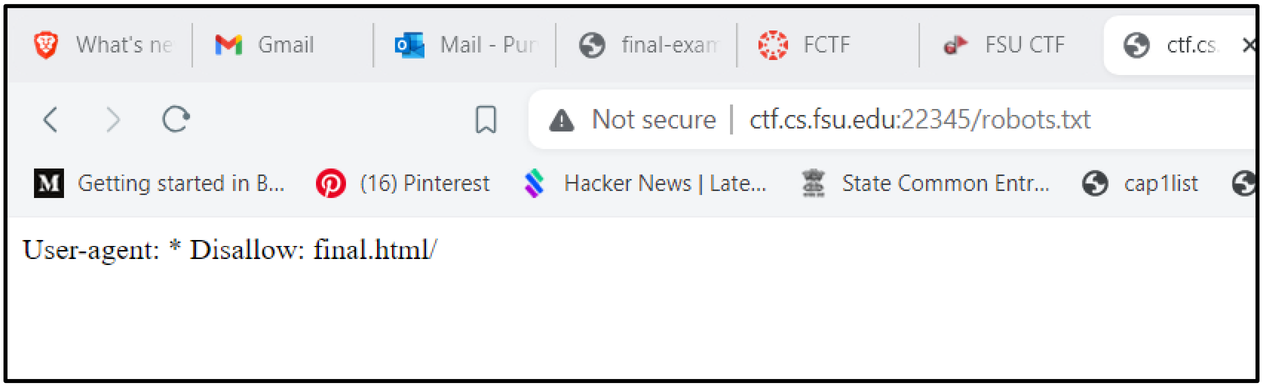
In this problem there is a webpage that has a sentence which involves the word robots in it.

That is how I tried of putting robots.txt in the url to open the robots page that every webpage consists of. It will have some information related to that particular webpage and that is how I got final.html page

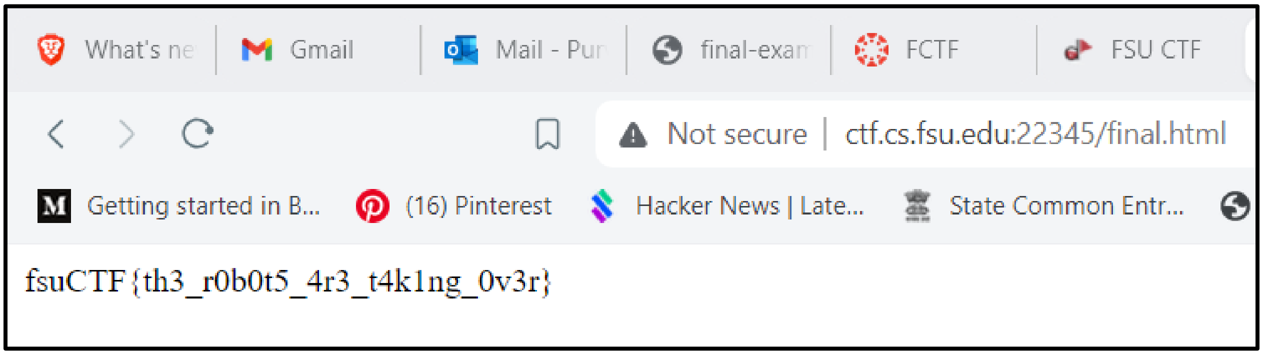
After entering the final.html in the url I got the flag.



The year is 2093 and this site is compromised by the new robots. Please save us.



User-agent: * Disallow: final.html/

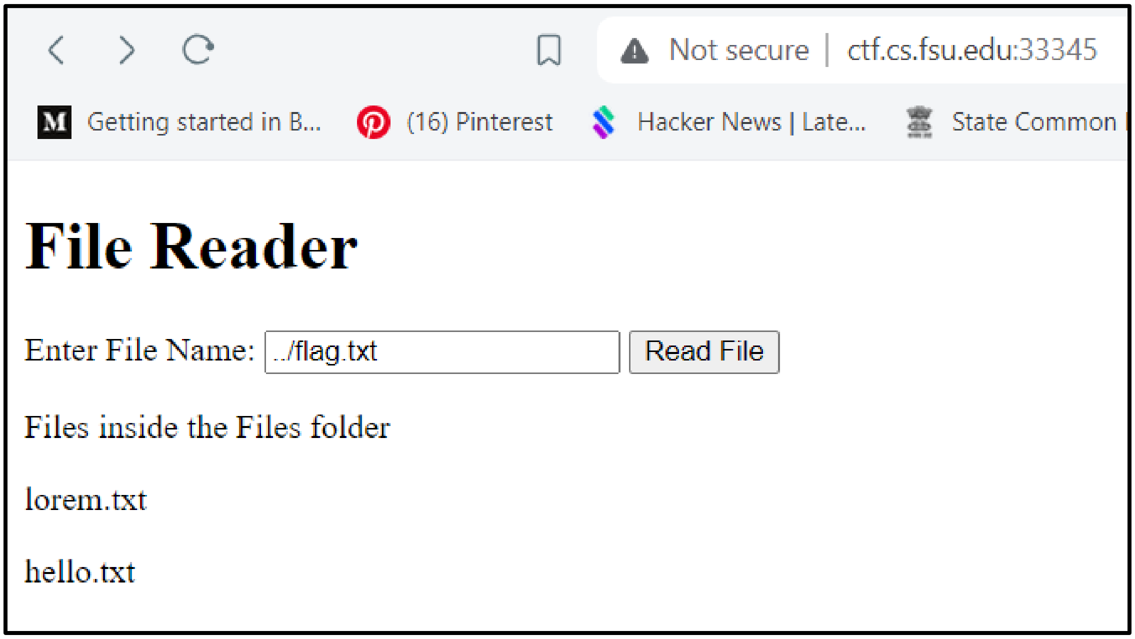


fsuCTF{th3_r0b0t5_4r3_t4k1ng_0v3r}

2. The path less traversed: fsuCTF{r34d1ng_r41nb0w}

In this problem the webpage consisted of a input box. And a few files were mentioned along with it.

Here I used the directory method and found my way back to the flags.txt with the help of the input box provided there.



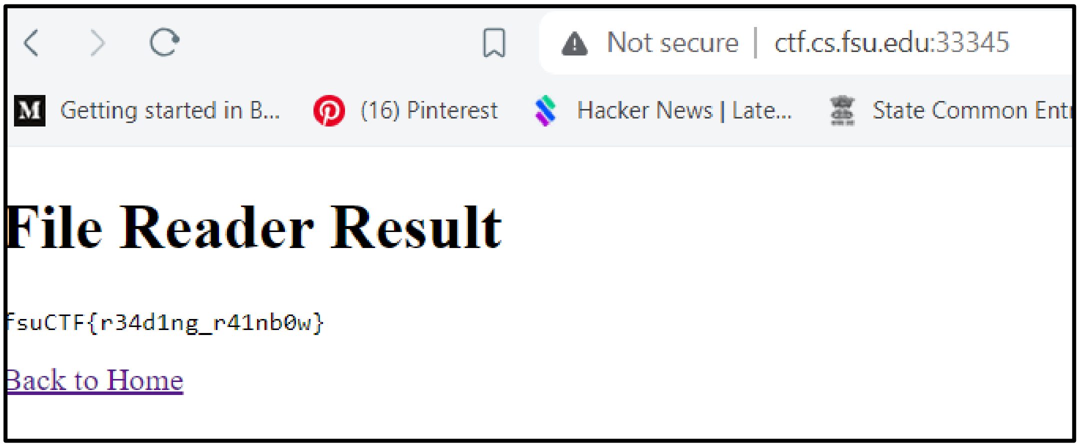
File Reader

Enter File Name: ../flag.txt

Files inside the Files folder

lorem.txt

hello.txt



File Reader Result

```
fsuCTF{r34d1ng_r41nb0w}
```

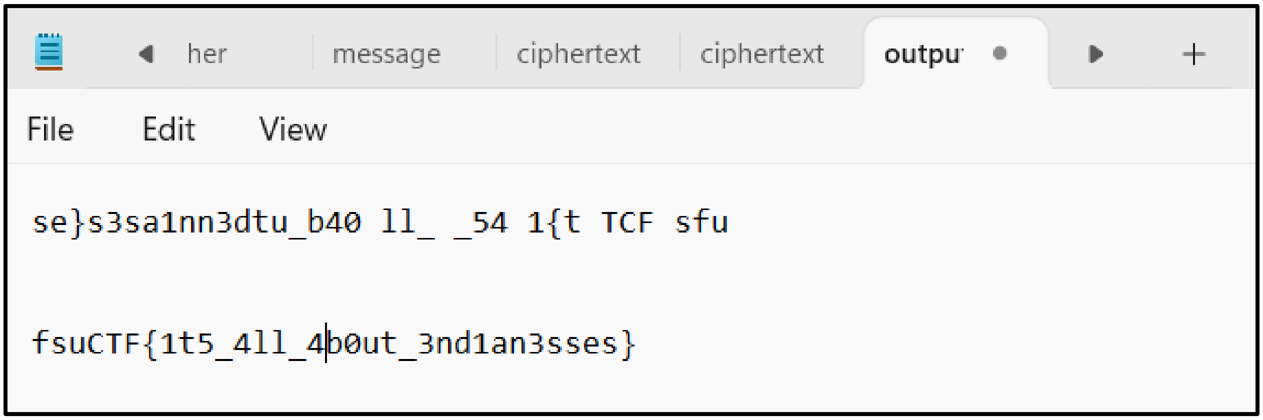
Back to Home

CRYPTO

1. **SHIFTY: fsuCTF{1t5_4ll_4b0ut_3nd1an3sses}**

In this problem there was a txt file provided which consisted of the jumbled flag.

So I figured out that to get the flag I had to arrange the letters in reverse as every 3 letters were shuffled with the middle letter coming first and then the letter left to it will come and then the word right to it will come. And it how I got the flag.



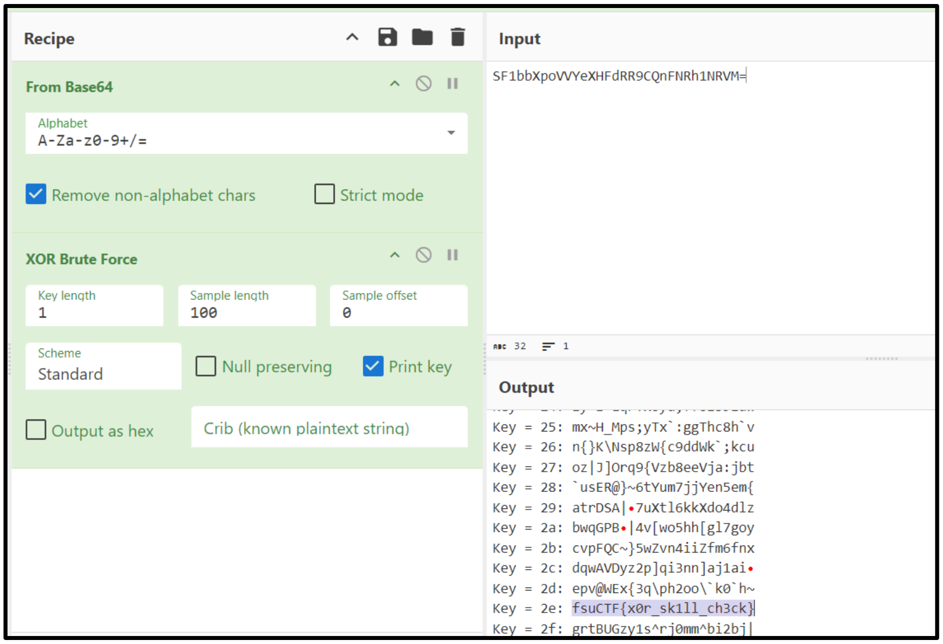
```
se}s3sa1nn3dtu_b40 ll_ _54 1{t TCF sfu


fsuCTF{1t5_4ll_4b0ut_3nd1an3sses}
```

2. **Skill issue: fsuCTF{x0r_sk1ll_ch3ck}**

In this problem there was a ciphertext that was provided. After analyzing it I figured out it a base64 encoding.

Using Cyberchef I decoded the string.

Again applied XOR bruteforce on the decoded string and that is how I got the flag.



Recipe

From Base64

Alphabet
A-Za-z0-9+/=

☑ Remove non-alphabet chars ☐ Strict mode

XOR Brute Force

Key length: 1 | Sample length: 100 | Sample offset: 0

Scheme: Standard ☐ Null preserving ☑ Print key

☐ Output as hex | Crib (known plaintext string)

Input

```
SF1bbXpoVVYeXHFdRR9CQnFNRh1NRVM=
```

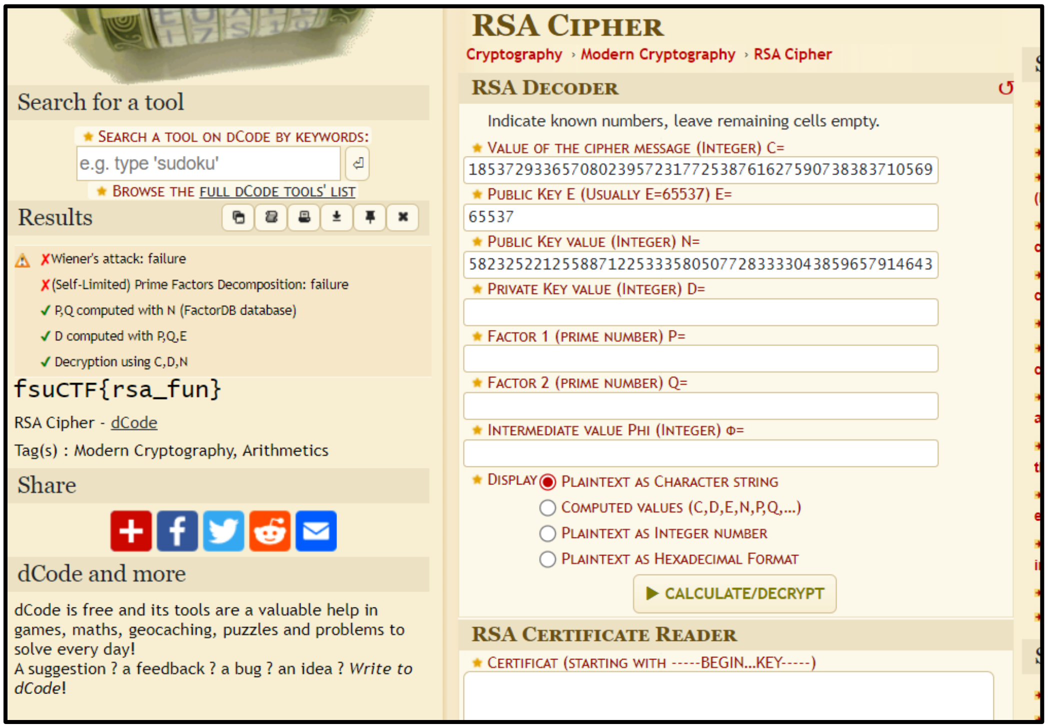
abc 32 ≡ 1

Output

```
Key = 25: mx~H_Mps;yTx`:ggThc8h`v
Key = 26: n{}K\Nsp8zW{c9ddWk`;kcu
Key = 27: oz|J]Orq9{Vzb8eeVja:jbt
Key = 28: `usER@}~6tYum7jjYen5em{
Key = 29: atrDSA|•7uXtl6kkXdo4dlz
Key = 2a: bwqGPB•|4v[wo5hh[gl7goy
Key = 2b: cvpFQC~}5wZvn4iiZfm6fnx
Key = 2c: dqwAVDyz2p]qi3nn]aj1ai•
Key = 2d: epv@WEx{3q\ph2oo\`k0`h~
Key = 2e: fsuCTF{x0r_sk1ll_ch3ck}
Key = 2f: grtBUGzy1s^rj0mm^bi2bj|
```

3. Really Secure Algorithm: fsuCTF{rsa_fun}

In this problem the txt consisted of the values of a RSA cipher. I simply put those values on the site. It didn't work so I changed the value of E to it default value that is always there for every problem and that is how I got the flag.



Search for a tool

★ Search a tool on dCode by keywords:

e.g. type 'sudoku'

★ Browse the full dCode tools' list

Results

✗ Wiener's attack: failure
✗ (Self-Limited) Prime Factors Decomposition: failure
✓ P,Q computed with N (FactorDB database)
✓ D computed with P,Q,E
✓ Decryption using C,D,N

fsuCTF{rsa_fun}

RSA Cipher - dCode

Tag(s) : Modern Cryptography, Arithmetics

Share

dCode and more

dCode is free and its tools are a valuable help in games, maths, geocaching, puzzles and problems to solve every day!
A suggestion ? a feedback ? a bug ? an idea ? *Write to dCode*!

RSA Cipher

Cryptography › Modern Cryptography › RSA Cipher

RSA Decoder

Indicate known numbers, leave remaining cells empty.

★ Value of the cipher message (Integer) C=
18537293365708023957231772538761627590738383710569

★ Public Key E (Usually E=65537) E=
65537

★ Public Key value (Integer) N=
5823252212558871225333580507728333043859657914643

★ Private Key value (Integer) D=

★ Factor 1 (prime number) P=

★ Factor 2 (prime number) Q=

★ Intermediate value Phi (Integer) Φ=

★ Display ● Plaintext as character string
○ Computed values (C,D,E,N,P,Q,...)
○ Plaintext as Integer number
○ Plaintext as Hexadecimal Format

▶ CALCULATE/DECRYPT

RSA Certificate Reader

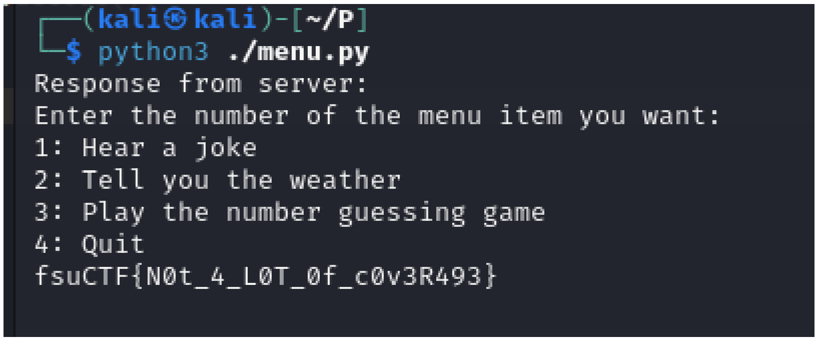
★ Certificat (starting with -----BEGIN...KEY-----)

PWN

1. **Menu: fsuCTF{N0t_4_L0T_0f_c0v3R493}**

In this problem I wrote a python code and that is how I got the flag.

```
import socket

target = ("ctf.cs.fsu.edu", 32333)

sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

sock.connect(target)

exploit_input = "A" * 100 + "\n"

sock.sendall(exploit_input.encode())

sock.shutdown(socket.SHUT_WR)

response = sock.recv(1024)

print("Response from server:")

print(response.decode())

sock.close()
```



```
┌──(kali㉿kali)-[~/P]
└─$ python3 ./menu.py
Response from server:
Enter the number of the menu item you want:
1: Hear a joke
2: Tell you the weather
3: Play the number guessing game
4: Quit
fsuCTF{N0t_4_L0T_0f_c0v3R493}
```

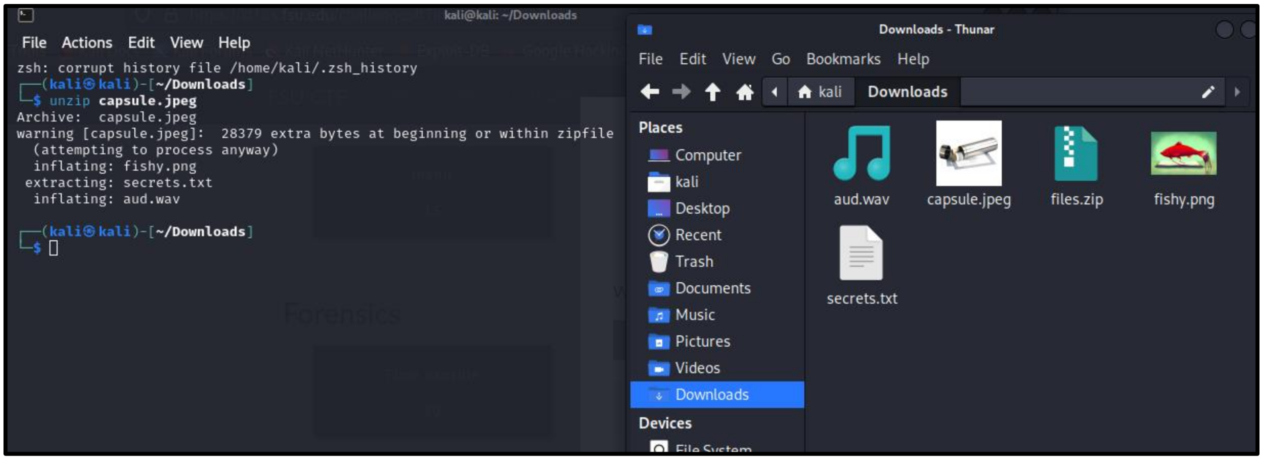
FORENSICS

1. **Capsule: fsuCTF{t!m3_c4psule5_c4n_h0ld_m4ny_s3cr3ts}**

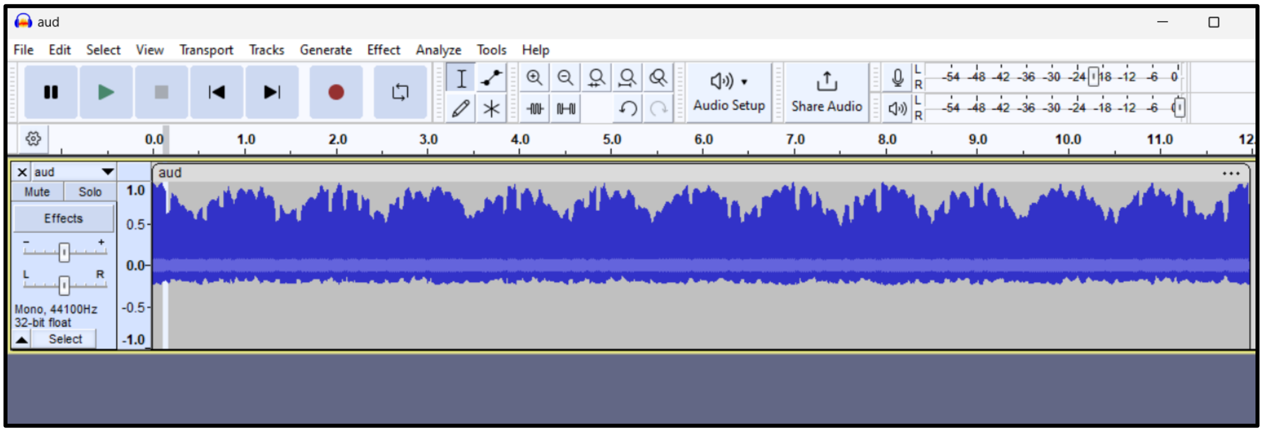
In this problem I used Kali. First I unzipped the file that was provided it was an image. After unzipping there were a few files they were to disguise.

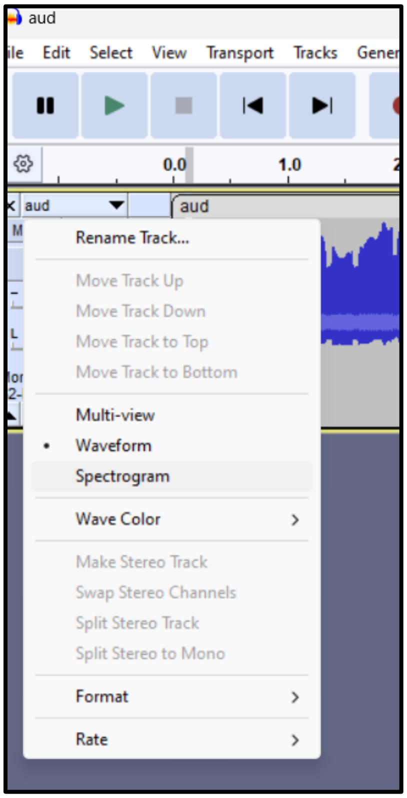
The main was audio file. I put this audio file in Audacity.

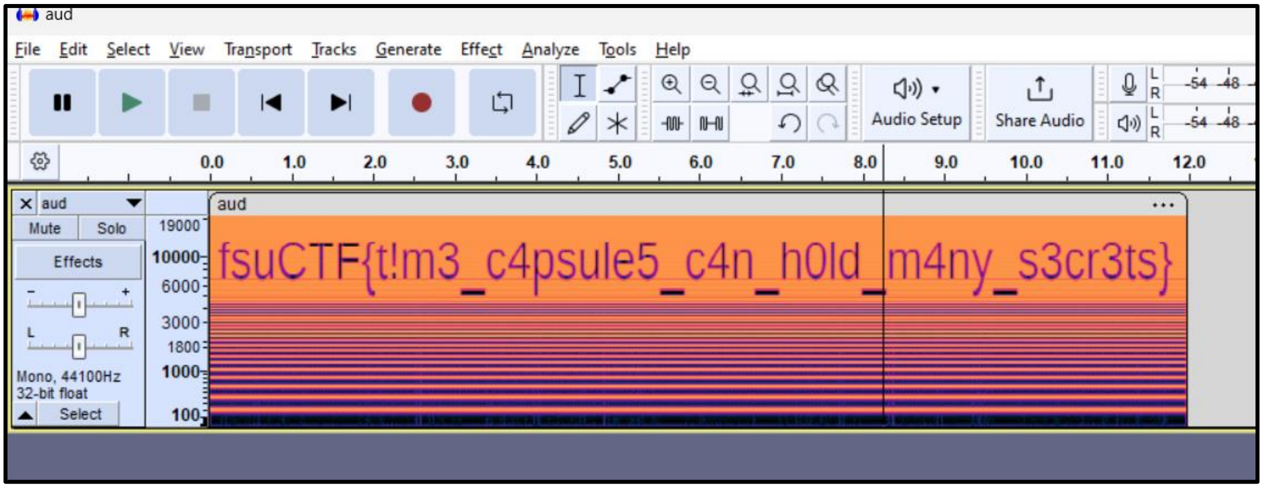
After applying spectrogram the waves produced the flag.



```
zsh: corrupt history file /home/kali/.zsh_history
┌──(kali㉿kali)-[~/Downloads]
└─$ unzip capsule.jpeg
Archive:  capsule.jpeg
warning [capsule.jpeg]:  28379 extra bytes at beginning or within zipfile
  (attempting to process anyway)
  inflating: fishy.png
 extracting: secrets.txt
  inflating: aud.wav

┌──(kali㉿kali)-[~/Downloads]
└─$ 
```





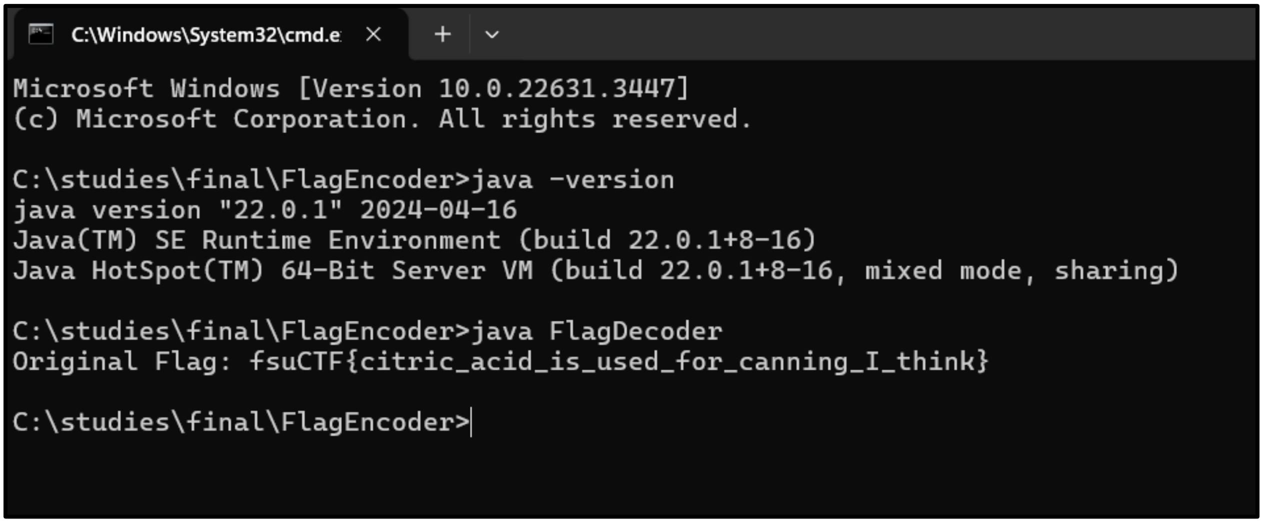


REVERSE ENGINEERING

1. **Mason Ball: fsuCTF{citric_acid_is_used_for_canning_I_think}**

In this problem the jar file was provided I simply extracted the file.

It consisted of a class file. I ran the program and that is how I got the flag.



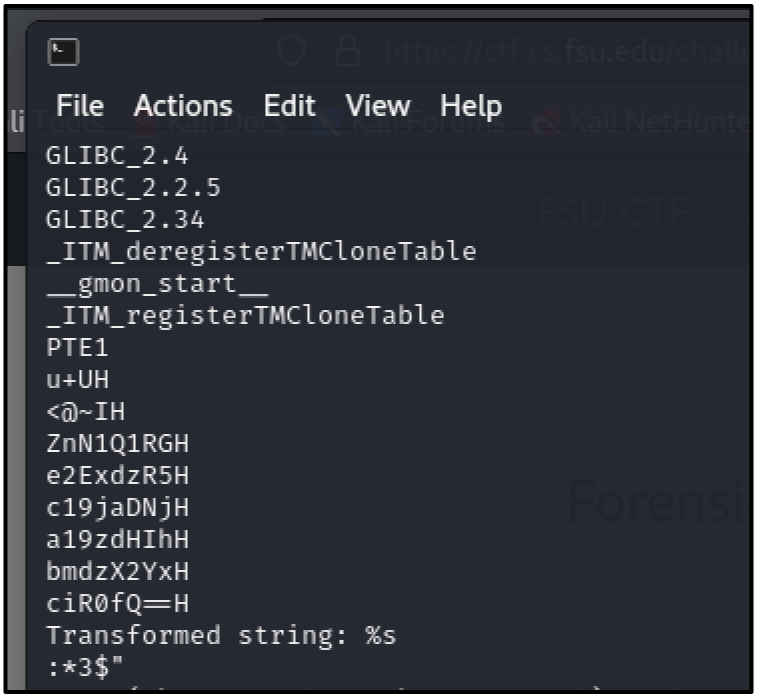
```
Microsoft Windows [Version 10.0.22631.3447]
(c) Microsoft Corporation. All rights reserved.

C:\studies\final\FlagEncoder>java -version
java version "22.0.1" 2024-04-16
Java(TM) SE Runtime Environment (build 22.0.1+8-16)
Java HotSpot(TM) 64-Bit Server VM (build 22.0.1+8-16, mixed mode, sharing)

C:\studies\final\FlagEncoder>java FlagDecoder
Original Flag: fsuCTF{citric_acid_is_used_for_canning_I_think}

C:\studies\final\FlagEncoder>
```

2. **Rev your engines: fsuCTF{a1w4ys_ch3ck_str!ngs_f1r$t}**

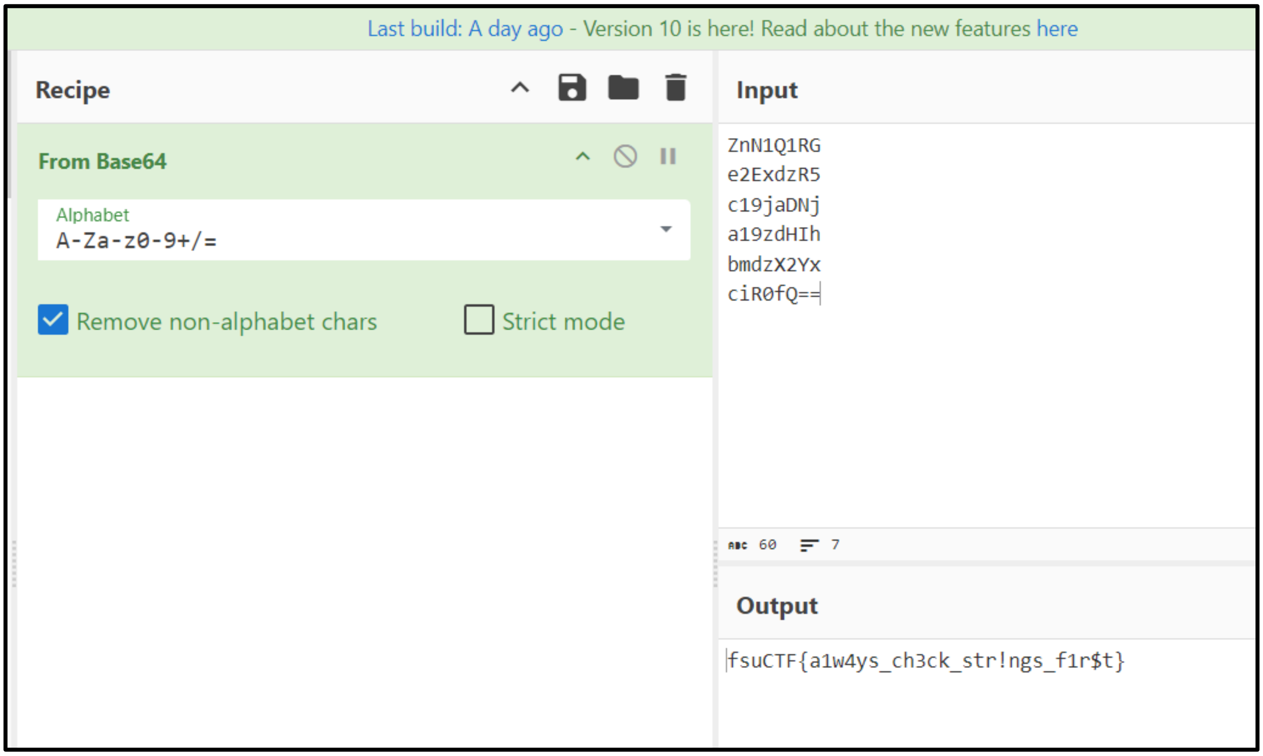
In this problem I ran the strings command on this file and got a list of strings.

After analyzing the strings. A group of them seemed like a base64 encoded string.

So I simply decoded them on Cyberchef and that is how I got the flag.



```
GLIBC_2.4
GLIBC_2.2.5
GLIBC_2.34
_ITM_deregisterTMCloneTable
__gmon_start__
_ITM_registerTMCloneTable
PTE1
u+UH
<@~IH
ZnN1Q1RGH
e2ExdzR5H
c19jaDNjH
a19zdHIhH
bmdzX2YxH
ciR0fQ==H
Transformed string: %s
:*3$"
```


Last build: A day ago - Version 10 is here! Read about the new features here
Recipe
Input
Output

From Base64

Alphabet
A-Za-z0-9+/=

☑ Remove non-alphabet chars   □ Strict mode

```
ZnN1Q1RG
e2ExdzR5
c19jaDNj
a19zdHIh
bmdzX2Yx
ciR0fQ==
```

abc 60   7

```
fsuCTF{a1w4ys_ch3ck_str!ngs_f1r$t}
```