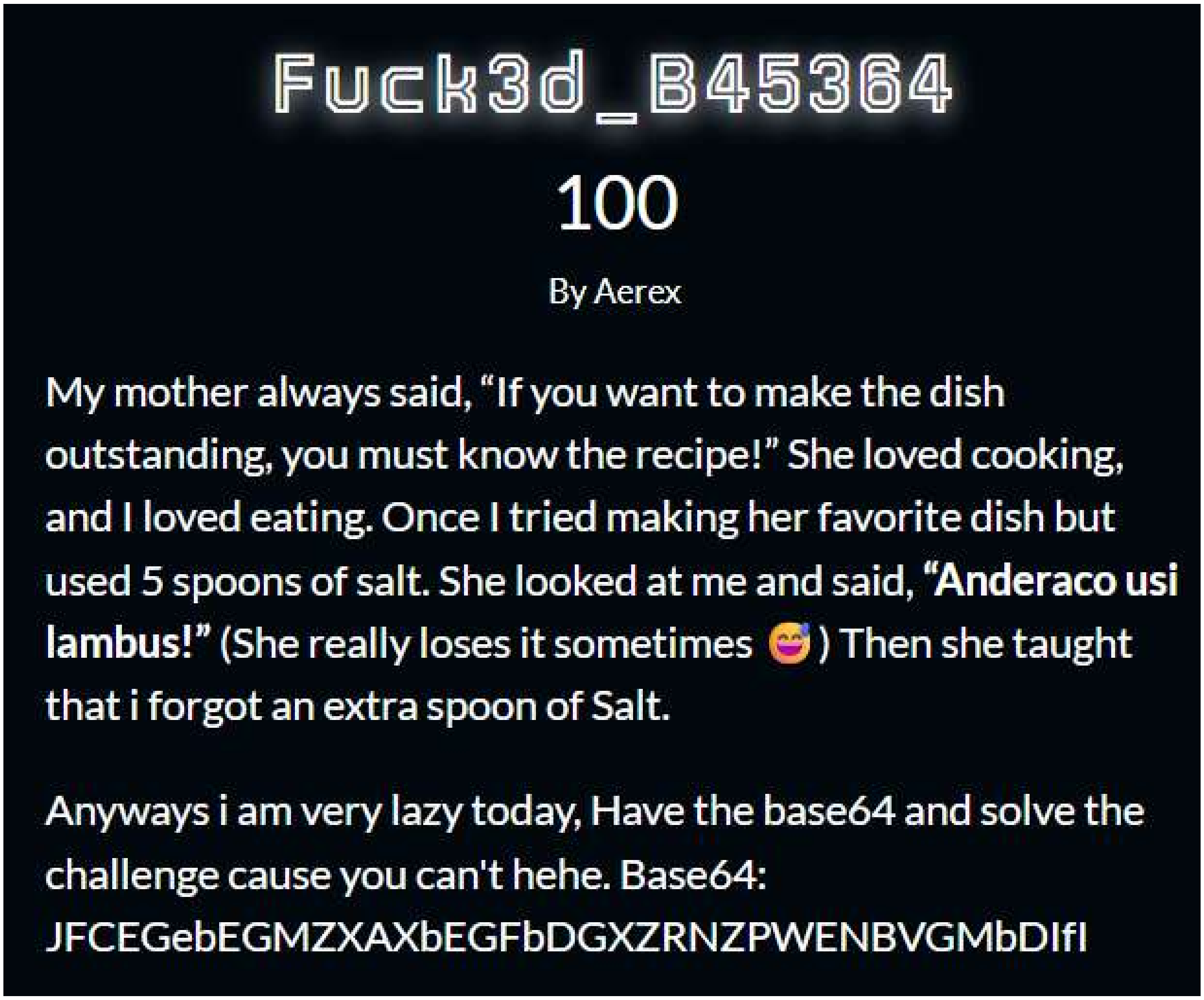
Fuck3d_B45364

07 November 2025 13:56

CATEGORY: Cryptography



Fuck3d_B45364

100

By Aerex

My mother always said, “If you want to make the dish outstanding, you must know the recipe!” She loved cooking, and I loved eating. Once I tried making her favorite dish but used 5 spoons of salt. She looked at me and said, **“Anderaco usi lambus!”** (She really loses it sometimes 😅) Then she taught that i forgot an extra spoon of Salt.

Anyways i am very lazy today, Have the base64 and solve the challenge cause you can't hehe. Base64: JFCEGebEGMZXAXbEGFbDGXZRNZPWENBVGMbDIfI

STEP BY STEP SOLUTION

A brief intro of how base64 encoding works:

So to encode a word like "hello" in base64 we first substitute the values of each character with the value assigned to them in ASCII

h --> 104--> 01101000
e --> 101--> 01100101
l --> 108--> 01101100
l --> 108--> 01101100
o --> 111--> 01101111

Then we concatenate all the bits together making it:

1101000 1100101 1101100 1101100 1101111

Then group it in groups of 6 (because 2^6 = 64):

011010 001100 101110 110011 011001 101111

Now we convert them back to characters using the base64 table.
i.e. A–Z → values 0–25, a–z → values 26–51, 0–9 → values 52–61, + → 62, / → 63, so we have:

011010 --> 26 --> a
001100 --> 12 --> M
101110 --> 46 --> u
110011 --> 51 --> z
011001 --> 25 --> Z
101111 --> 47 --> v

Thus "hello" becomes "aMuzZv" in base64.

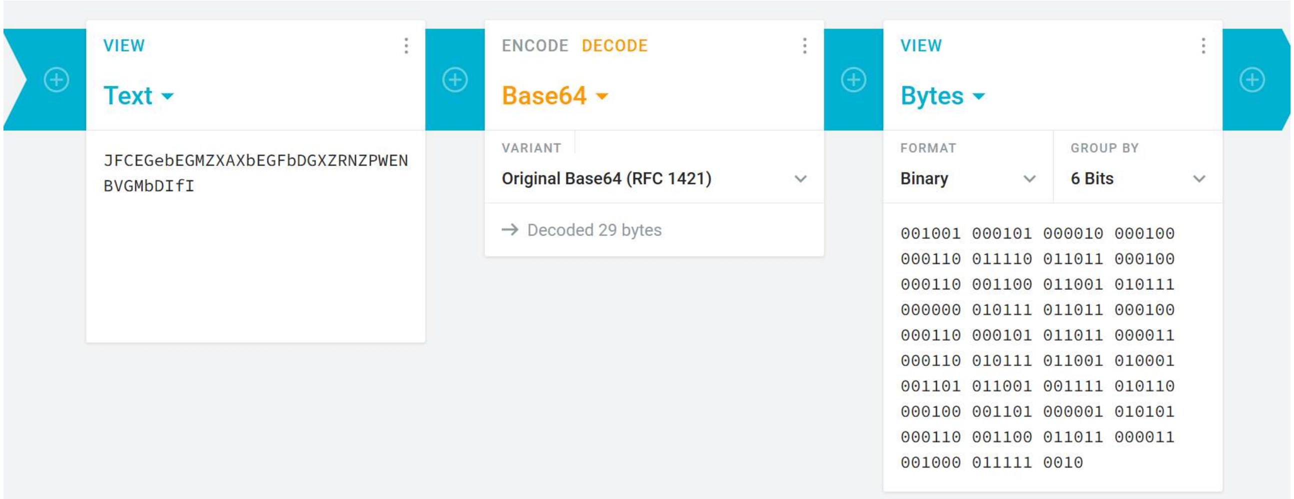
Now in the challenge it is given the author used "5 spoons of salt" and "forgot an extra spoon of salt". (according to the author's mother)
Each spoon here refers to a bit while encoding in base64. This is because we group the resulting binary form of the text in groups of 6. (the six spoons required).

So the author must have grouped it in groups of 5 instead of 6 and then looked up the values in the base64 table to encode it.

(NOTE: the encoding DOES NOT contain even a single number and is moreover dominated by capital letters. More specifically only characters corresponding to 0 to 31 in base64 are present in the encoded text)

So now that we know how the encoding was done our workflow would be as follows to retrieve the flag:

1. Convert the given encoded text to binary in groups of 6. We note that the first digit of each group is always 0 and was NOT present when the grouping was done.



```
JFCEGebEGMZXAXbEGFbDGXZRNZPWEN
BVGMbDIfI
```

VARIANT: Original Base64 (RFC 1421)

→ Decoded 29 bytes

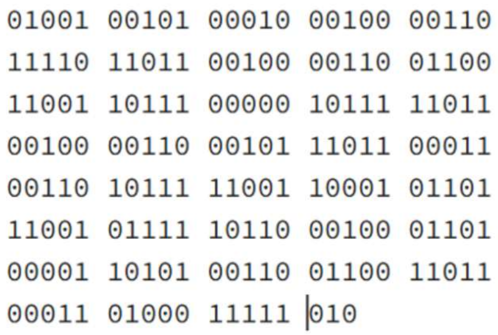
FORMAT: Binary

GROUP BY: 6 Bits

```
001001 000101 000010 000100
000110 011110 011011 000100
000110 001100 011001 010111
000000 010111 011011 000100
000110 000101 011011 000011
000110 010111 011001 010001
001101 011001 001111 010110
000100 001101 000001 010101
000110 001100 011011 000011
001000 011111 0010
```

<https://cryptii.com/pipes/base64-to-binary>

2. We remove all the redundant zeroes at the starting to get the original text in binary.



```
01001 00101 00010 00100 00110
11110 11011 00100 00110 01100
11001 10111 00000 10111 11011
00100 00110 00101 11011 00011
00110 10111 11001 10001 01101
11001 01111 10110 00100 01101
00001 10101 00110 01100 11011
00011 01000 11111 010
```

3. We now have the flag but it is in binary. To get the flag we group the bits in groups of 8 and convert back to text using ASCII

FLAG: IDC{d33p_d1v3_1n_b45364}