first challenge: What is your favorite dish?

S1hfRldJe0ZESFZEVV9WRE9ER19MVl9XS0hfRUhWV30=

Sol:

Notice the == at the end of the cipher this is mean base 64 (“not always”) ,so we decrypted it and got this second cipher : https://www.base64decode.org/

KX_FWI{FDHVDU_VDODG_LV_WKH_EHVW}

And according to the title and the format we could think of Caesar, so the result is:

https://www.dcode.fr/caesar-cipher

HU_CTF{CAESAR_SALAD_IS_THE_BEST}

Second challenge: Weird Vigenere

Given an cipher.txt When we try to google this sequence "dah-di-dit" we found that simbols is used for Morse code encryption, for example "A" is "di-dah", "B" is "dah-di-di-dit" etc. decrypt it by writing python script to translate it.

```
#!/usr/bin/env python3

# -*- coding:utf-8 -*-

#lang => https://morsecode.scphillips.com/morse.html

lang = {'Di-dah':'A','Dah-di-di-dit':'B','Dah-di-dah-dit':'C','Dah-di-dit':'D','Dit':'E','Di-di-dah-dit':'F','Dah-dah-dit':'G','Di-di-di-dit':'H','Di-dit':'I','Di-dah-dah-dah':'J','Dah-di-dah':'K','Di-dah-di-dit':'L','Dah-dah':'M','Dah-dit':'N','Dah-dah-dah':'O','Di-dah-dah-dit':'P','Dah-dah-di-dah':'Q','Di-dah-dit':'R','Di-di-dit':'S','Dah':'T','Di-di-dah':'U','Di-di-di-dah':'V','Di-dah-dah':'W','Dah-di-di-dah':'X','Dah-di-dah-dah':'','Dah-dah-di-dit':'Z','Dah-dah-dah-dah-dah':'0','Di-dah-dah-dah-dah':'1','Di-di-dah-dah-dah':'2','Di-di-di-dah-dah':'3','Di-di-di-di-dah':'4','Di-di-di-di-dit':'5','Dah-di-di-di-dit':'6','Dah-dah-di-di-dit':'7','Dah-dah-dah-di-dit':'8','Dah-dah-dah-dah-dit':'9','Di-dah-di-dah':'ä','Di-dah-dah-di-dah':'á','Di-dah-dah-di-dah':'å','Dah-dah-dah-dah':'Ch','Di-di-dah-di-dit':'é','Dah-dah-di-dah-dah':'ñ','Dah-dah-dah-dit':'ö','Di-di-dah-dah':'ü','Di-dah-di-di-dit':'&','Di-dah-dah-dah-dah-dit':'\'','Di-dah-dah-di-dah-dit':'@','Dah-di-dah-dah-di-dah':')','Dah-di-dah-dah-dit':'(','Dah-dah-dah-di-di-dit':':','Dah-dah-di-di-dah-dah':',','Dah-di-di-di-dah':'=','Dah-di-dah-di-dah-dah':'!','Di-dah-di-dah-di-dah':'.','Dah-di-di-di-di-dah':'-','Di-dah-di-dah-dit':'+','Di-di-dah-dah-di-dit':'?','Dah-di-di-dah-dit':'/'}

flag = "Dah-dah-dah-dah-dah Dah-di-di-dah Di-di-di-di-dah Dah-di-dah-dit Dah-di-di-di-dit Di-di-dah-dit Di-di-di-dah-dah Dah-dah-dah-dah-dah Dah-di-di-di-dit Dah-di-di-dit Di-di-dah-dah-dah Di-dah Dah-dah-di-di-dit Di-di-di-di-dit Di-di-di-di-dit Dah-dah-dah-dah-dah Di-di-dah-dah-dah Di-dah Di-di-di-di-dah Dah-dah-di-di-dit Di-di-di-dah-dah Di-di-di-dah-dah Dah-dah-di-di-dit Di-di-di-di-dah Di-di-di-dah-dah Dit Di-di-di-di-dah Dah-di-di-dit Di-di-di-dah-dah Di-di-di-dah-dah Dah-dah-di-di-dit Dah-dah-dah-dah-dit Di-di-dah-dah-dah Di-dah-dah-dah-dah Di-di-dah-dah-dah Di-dah-dah-dah-dah Di-di-dah-dah-dah Di-dah-dah-dah-dah Di-di-di-di-dah Di-di-di-dah-dah Di-di-di-dah-dah Di-di-di-di-dah Di-di-di-di-dit Di-di-di-di-dit Di-di-di-dah-dah Di-di-di-di-dit Di-di-di-di-dah Di-di-di-di-dit Di-di-dah-dah-dah Di-di-di-di-dit Di-di-dah-dah-dah Di-di-di-di-dit Di-di-di-dah-dah Di-di-di-di-dit Di-di-di-di-dit Di-di-di-di-dah Di-di-di-di-dit Di-di-di-di-dit Di-di-di-di-dit Dah-dah-dah-dah-dah Di-di-di-dah-dah Di-dah-dah-dah-dah Di-di-di-di-dah Di-di-di-di-dah Di-di-di-dah-dah Di-dah-dah-dah-dah Dah-dah-di-di-dit Di-di-di-di-dah Dah-dah-di-di-dit Dah-dah-dah-dah-dit Di-di-dah-dah-dah Di-dah Di-di-di-dah-dah Di-dah-dah-dah-dah Di-di-di-dah-dah Di-di-di-di-dit Di-di-dah-dah-dah Di-di-di-dah-dah Dah-di-di-di-dit Dah-dah-dah-di-dit Di-di-di-dah-dah Di-di-di-di-dah Di-di-di-dah-dah Di-di-di-di-dit Di-di-di-di-dah Dah-dah-dah-di-dit Dah-dah-di-di-dit Di-di-di-di-dah Di-di-di-dah-dah Di-di-di-di-dah Di-di-di-di-dah Dah-dah-di-di-dit Di-di-dah-dah-dah Di-di-di-dah-dah Di-di-dah-dah-dah Di-dah-dah-dah-dah Di-di-di-di-dit Di-di-di-di-dah Di-di-di-dah-dah Di-di-dah-dah-dah Di-di-di-dah-dah Di-di-di-dah-dah Di-di-di-di-dah Dit Dah-di-di-di-dit Di-di-di-
```

```
dah Di-di-di-dah-dah Di-dah-dah-dah-dah Dah-di-di-di-dit Dit Di-di-di-di-dah Dah-dah-di-di-dit Di-di-di-dah-dah Dah-di-di-dit Di-di-dah-dah-dah Dah-dah-dah-dah-dit Di-di-dah-dah-dah Dah-dah-dah-dah-dah Di-di-di-di-dit Dah-dah-dah-dah-dah Di-di-di-di-dah Dah-di-di-dit Di-di-di-di-dit Di-di-dah-dit Di-di-di-di-dah Dah-dah-di-di-dit Di-di-di-di-dit Dah-dah-dah-dah-dit Di-di-di-di-dit Dah-dah-dah-dah-dit Dah-dah-di-di-dit Dah-di-di-dit Di-di-di-di-dah Di-di-di-di-dah Di-di-di-di-dit Dah-dah-dah-di-dit Di-di-di-di-dah Di-dah-dah-dah-dah Di-di-di-di-dit Di-dah Di-di-di-di-dit Di-di-dah-dit Di-di-di-di-dit Dah-di-di-di-dit Di-di-di-di-dah Di-di-di-di-dit Di-di-di-di-dit Di-di-dah-dit Di-di-di-di-dah Dah-dah-di-di-dit Di-di-di-di-dit Di-di-di-di-dit Di-di-di-di-dit Dah-dah-dah-di-dit Di-di-di-di-dah Dah-di-dit Di-di-di-di-dit Di-di-dah-dit Di-di-di-di-dah Di-dah-dah-dah-dah Di-di-di-di-dah Di-di-di-di-dit Di-di-di-di-dit Di-di-dah-dah-dah Dah-dah-di-di-dit Dah-di-dit Di-di-di-di-dah Dah-dah-dah-dah-dah Dah-di-di-di-dit Di-di-di-dah-dah Dah-di-di-di-dit Di-di-dah-dit Dah-dah-di-di-dit Di-di-dah-dah-dah Dah-di-di-di-dit Di-di-dah-dit Dah-di-di-di-dit Dit Dah-di-di-di-dit Di-dah-dah-dah-dah Dah-di-di-di-dit Dah-di-di-di-dit Dah-di-di-di-dit Dah-di-dah-dit Dah-di-di-di-dit Di-dah-dah-dah-dah Dah-di-di-di-dit Dah-dah-di-di-dit"

#print(flag)

#tmp - morse words

morse = []

word_tmp = ''

translate = ''

#remove the space from the string

morse = flag.split()

#print(morse)

for k in range(0,len(morse)):

    if morse[k] in lang:

        translate+=lang[morse[k]]

    else:

        pass

#print(translate)

hex_string = translate[2:]

bytes_object = bytes.fromhex(hex_string)

ascii_string = bytes_object.decode("ASCII")


print(ascii_string)
```

then the result will be :

Lo0k*uP*G3t>K3y!!!C4U5E%%5TUP1D1ty*15#h45Ht4G#!T23Nd1nG;)
PK_GYY{DXAZ_VE_GUXM_AER}@coronaflag

Still we need more work to get the flag. It is encrypted by Beaufort Cipher as the title suggest by this website https://www.dcode.fr/beaufort-cipher and the key as the decrypted text said look up get key it’s weird or we could use KNOWING A PLAINTEXT WORD:HU_CTF, then we know the key is weird use it this time to decrypt the cipher but WITH THE CIPHER KEY:WEIRD, to get the wanted format. the result will be: HU_CTF{THIS_IS_YOUR_WAR}

Third challenge: Gg RSA

n=309490746288788399022959624251210450709632544651034698988190913831674454215104329115656687

e=65537

c=56424018463638628931875913164641091675064027238426767825847448892221931719430604017834655

we have first to factorize n into p,q into this link:

http://factordb.com/index.php?query=309490746288788399022959624251210450709632544651034698988190913831674454215104329115656687

then write a short script in python to get the flag:

```
#!/usr/bin/env python


from Crypto.Util.number import inverse

n=309490746288788399022959624251210450709632544651034698988190913831674454215104329115656687

e=65537

c= 56424018463638628931875913164641091675064027238426767825847448892221931719430604017834655

p=257023740566127971049505284188227650127587089

q=1204132916309968387505657927692462380137494783

phi = ( p-1 )* ( q - 1 )


d = inverse(e,phi)


m = pow( c, d, n )


print hex(m)[2:-1].decode('hex')
```

then the flag is HU_CTF{3A5Y_P3A5Y_L3M0N_5QU3EZY}

Fourth challenge: Oops I did it again!

This challenge come with picture that the last two digits of the key and the IV and part of the cipher text is unknown so we have first to Get the AES key

With this script

```
from Crypto.Cipher import AES
from operator import xor
import binascii, sys


KEY_first = "C313F4_R31E4H6" # brute force the key's last two characters
cipher1 = "1C00000000000000000000000000CD91" #padding to unknown #in this algorithm we use aes-128 that means 128bit
cipher2 = "FC05A9D10789518428975FDC93D82CC2" #32byte length of each cipher block
plain1 = "The message is p"#16byte plaintext block
plain2 = "rotected by AES!"


def decrypt(cipher, passphrase):
   aes = AES.new(passphrase, AES.MODE_CBC, binascii.unhexlify(cipher1))        # cipher = AES.new(self.key, AES.MODE_CBC, iv )

   return aes.decrypt(cipher)


# iterate through relavent ascii range
for i in range(32, 126):
   for j in range(32, 126):
      key = KEY_first + chr(i) + chr(j)
      dec_plain2 = decrypt(binascii.unhexlify(cipher2),  key)
      if  str(dec_plain2).startswith("r") and str(dec_plain2).endswith('S!'):
         print "decrypted plain2: " + dec_plain2 + " with key: " + key
```

the output is C313F4_R31E4H613

second we have to Get the first ciphertext block by this script

```
from Crypto.Cipher import AES
import binascii, sys
```

```
KEY = "C313F4_R31E4H613"
plain2 = "rotected by AES!"

cipher2= "FC05A9D10789518428975FDC93D82CC2"

def decrypt(cipher,passphrase):
   aes = AES.new(passphrase,AES.MODE_CBC,plain2)
   return aes.decrypt(cipher)

# Output result
print "Decrypted data: " + binascii.hexlify(decrypt(binascii.unhexlify(cipher2), KEY))
```

output: 1CB9CD2E744B127793A2C868EFCDCD91

third: Get the IV by this script:

```
from Crypto.Cipher import AES
import binascii, sys

KEY="C313F4_R31E4H613"
IV="The message is p"

cipher1="1cb9cd2e744b127793a2c868efcdcd91"

def decrypt(cipher,passphrase):
   aes = AES.new(passphrase,AES.MODE_CBC,IV)
   return aes.decrypt(cipher)

print "decrypted data: " + decrypt(binascii.unhexlify(cipher1), KEY)
```

the result will be HU_CTF{#H0RR8L3}