h@cktivitycon

hexahedron

Description

"So much of crypto is all about shapes! Since some shapes have so many special sides :)"

We get a file, hexahedron.txt to download. When we open it, we see:

```
n =
0x9ffa2a58ad286990fc5fe97b669e8cb2752e81fafa5ac774ea856d8ca124089ba4b06fe21a5d588c1dcb9602838d

e = 0x3
c =
0x10652cdfaa6a6f6f688b98219cd32ce42c4d4df94afaea31cd94dfac50678b1f50f3ab1fd389f9998b6727ffd1a2c
```

OK it looks like some sort of an RSA challenge. First thing I notice is that the n, e and c are in hex. Normally for these length integers I would do it from the command line in python, however in my travels I have a small script taken from a Microsoft site that converts using int()

```
# Python3 code to demonstrate
# converting hexadecimal string to decimal

# Using int()

# initializing string
test_string =
'0x10652cdfaa6a6f6f688b98219cd32ce42c4d4df94afaea31cd94dfac50678b1f50f3ab1fd389f9998b6727ffd1a2c


# printing original string
print("The original string : " + str(test_string))

# using int()
# converting hexadecimal string to decimal
res = int(test_string, 16)

# print result
print("The decimal number of hexadecimal string : " + str(res))
```

With the result below shown for c :

```
The original string :
0x10652cdfaa6a6f6f688b98219cd32ce42c4d4df94afaea31cd94dfac50678b1f50f3ab1fd389f9998b6727ffd1a2c


The decimal number of hexadecimal string :
22173447507981785996165188818512381920465373711348319848288944137525209373781614868802699
```

```
>>>
```

Now I have all three, I can use Genapati's RsaCtfTool.

```
/RsaCtfTool.py -n
11233981630192539692621128968979374581421392531427388607130578587417802855251048223903653
 -e 3 --uncipher
22173447507981785996165188818512381920465373711348319848288944137525209373781614868802699
```

I won't print off all the attacks but needless to say a few minutes later we get the flag.

```
[*] Performing pollard_rho attack on /tmp/tmprao1ktzw.
[!] Timeout.
[*] Performing londahl attack on /tmp/tmprao1ktzw.
100%|████████████████████| 20000001/20000001 [00:09<00:00, 2014348.14it/s]
100%|████████████████████| 20000001/20000001 [00:17<00:00, 1140050.25it/s]
[*] Performing binary_polinomial_factoring attack on /tmp/tmprao1ktzw.
[*] Performing euler attack on /tmp/tmprao1ktzw.
[!] Timeout.
[*] Performing smallfraction attack on /tmp/tmprao1ktzw.
[*] Performing cm_factor attack on /tmp/tmprao1ktzw.
100%|██████████████████████████████████| 6/6 [00:00<00:00,
12.99it/s]

Unciphered data :
HEX : 0x666c61677b3038306561656230643866373234626362353432353632623362623730386535
7d
INT (big endian) :
13040004482819483500695568095911430044119005002403329881609730936713446088302388627030488

INT (little endian) :
15940898332589092471641868064009868552823996352415667103914137547565850266448879169635970

STR : b'flag{080eaeb0d8f724bcb542562b3bb708e5}'
```

Flag:
flag{080eaeb0d8f724bcb542562b3bb708e5}