Mark Seliternikov
picoCTF - miniRSA [300 points]

```
┌──(kali㉿kali)-[~/Desktop]
└─$ cat ciphertext

N: 29331922499794985782735976045591164936683059380558950386560160105740343201513369939006307531165922708949619162698623675349030430859547825708994708321803705309459438099340427770580064400911431856656901982789948285309956111848686906152664473350940486507451771223435835260168971210087470894448460745593956840586530527915802541450092946574694809584880896601317519794442862977471129319781313161842056501715040555964011899589002863730868679527184420789010551475067862907739054966183120621407246398518098981106431219207697870293412176440482900183550467375190239898455201170831410460483829448603477361305838743852756938687673
e: 3

ciphertext (c): 22053164139311340310746037469282477990301552212525198726500730107820491798569760805127162373088822942263693004127199959040649318195314563929579571224596407364240897447722219335008609363314592808322114455483324293385723698237047846253689 33
```

(לאתגר זה החלטתי לנסות להשתמש ב-Kali)

לאתגר זה מקבלים טקסט מוצפן + המפתח הציבורי + המקדם N.

לפי הנתונים אפשר למצוא את הטקסט המוצפן בגלל שה-e ממש קטן (3).
אפשר לעשות זאת מכיוון שהתוצאה של הטקסט תהיה ממש קטנה יחסית ל-N. (מבחינתי הוא יכול להיות אינסופי).
מה שאומר שאני יכול להתעלם ממנו לגמרי.

```
# mark's solution :)

from math import pow, trunc

ciphertext = 22053164139311340310746037469282477990301552212525198726500730107820

int_text = trunc(pow(ciphertext, 1./3.))

print(f"plaintext in int : {int_text}\n")

hex_text = hex(int_text)

print(f"plaintext in hex : {hex_text}\n")
```

לכן מה שעשיתי היה למצוא את השורש בשלישית של הטקסט המוצפן

```
┌──(kali㉿kali)-[~/Desktop]
└─$ python3 solver.py
plaintext in int : 13016382529448975049937056264654191108020998208395813341903515045258909345382400

plaintext in hex : 0x7069636f4353080000000000000000000000000000000000000000000000000000000000000000
```

אך לצערי זה לא היה מספיק מדויק, לא הצלחתי למצוא את כל הדגל. (אבל לפחות אני יודע שאני בכיוון) :)

Text Converter

7069636f4353080000000000000000
0000000000000000000000000000
0000000000000

picoCS

```
def root3rd(x):
    y, y1 = None, 2
    while y≠y1:
        y = y1
        y3 = y**3
        d = (2*y3+x)
        y1 = (y*(y3+2*x)+d//2)//d
    return y

int_text = trunc(root3rd(ciphertext))

print(f"find_invpow result : {int_text}\n")

hex_text = hex(int_text)

print(f"hex_text : {hex_text}\n")
```

לחיפשתי פונקציה מוכנה שתמצא לי את השורש בשלישית הכי מדוייק שאפשר.
מצאתי ב-: GitHub
https://github.com/JohnHammond/ctf-katana#cryptography

ממש נוח, יש שם פונקציה מוכנה למקרה ש-e הוא 3.

```
find_invpow result : 13016382529449106065894479374027604750406953699090365388203708028670029596145277

hex_text : 0x7069636f4354467b6e3333645f615f6c41726733725f655f63636161373737367d
```

עכשיו כאשר בדקתי התוצאה הייתה הרבה יותר הגיונית

עכשיו כל מה שנשאר לעשות זה לבדוק אם התוצאה הייתה מדוייקת מספיק.

כן, זה אכן הדגל :)