Mark Seliternikov
picoCTF - rsa pop quiz [200 points]

לאתגר זה מקבלים הדרכה להשתמש ב-NetCat
ולהקשיב לכתובת ופורט.

```
jupiter.challenges.picoctf.org 58617
```

לאחר שמתחברים מקבלים את הפלט הבא.
ניתן לראות שזה בוחן לידע RSA.

```
mark@XPS-13-9360:~$ nc jupiter.challenges.picoctf.org 58617
Good morning class! It's me Ms. Adleman-Shamir-Rivest
Today we will be taking a pop quiz, so I hope you studied. Cramming just will no
t do!
You will need to tell me if each example is possible, given your extensive crypt
o knowledge.
Inputs and outputs are in decimal. No hex here!
#### NEW PROBLEM ####
q : 60413
p : 76753
##### PRODUCE THE FOLLOWING ####
n
IS THIS POSSIBLE and FEASIBLE? (Y/N):
```

בבוחן זה יש מספר שאלות, כתבתי סקריפט בפייתון
שעונה על כל שאלות הבוחן (גם לשאלות שהתשובה
שלהן היא n/no).

העיקרון זה לדעת אם השאלות ניתנות לפתור.
למשל: שאלה האם אפשר לפענח הצפנה בעזרת e.
התשובה היא לא מכיוון שהוא ה-public key.
כלומר, הוא ה-encryptor.
כדי לענח הצפנה צריך את d, כלומר את הדקריפטור.

```python
print("Question 1")

q = 60413
p = 76753

print(f"n : {q * p}")

print("Question 2")

p = 54269
n = 5051846941

print(f"q : {n // p}")

print("Question 3")
print("n")

print("Question 4")

q = 66347
p = 12611

print(f"tointent(n) : {(q - 1) * (p - 1)}")

print("Question 5")

plaintext = 635729417148931154719098761554457513358196788649
e = 3
n = 291294636093263225595211231362220787805854512081491385 47

c = (plaintext ** e) % n

print(f"ciphertext : {c}")

print("Question 6")
print("n")

def egcd(a, b):
    if a == 0:
        return (b, 0, 1)
    else:
        g, y, x = egcd(b % a, a)
        return (g, x - (b // a) * y, y)

def modinv(a, m):
    g, x, y = egcd(a, m)
    if g != 1:
        raise Exception('modular inverse does not exist')
    else:
        return x % m

q = 92092076805892533739724722602668675840671093008520241548
p = 97846775312392801037224396977012615848433199640105786119
e = 65537

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

d = modinv(e, phin)

print(f"d : {d}")
```

```python
print("Question 7")

p = 15314304227252786879841261241720443415693514687428299094
ciphertext = 15898528329836486259454280642537682745047492256
e = 65537
n = 23952937352643527451379227516428377705004894508566304311

q = n // p

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

d = modinv(e, phin)

m = pow(ciphertext, d, n)

print(f"plaintext : {m}")
```

```
IS THIS POSSIBLE and FEASIBLE? (Y/N):y
#### TIME TO SHOW ME WHAT YOU GOT! ###
plaintext: 14311663942709674867122208214901970650496788151239520971623411712977119660454037678408741501
Outstanding move!!!


If you convert the last plaintext to a hex number, then ascii, you'll find what you need! ;)
```

לאחר שפותרים את השאלה האחרונה הפלט שמקובל הוא ciphertext אשר מוצפן בASCII וב-HEX

```
Python 3.8.5 (default, Jul 28 2020, 12:59:40)
[GCC 9.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> hex(14311663942709674867122208214901970650496788151239520971623411712977119660454037678408741501)
'0x7069636f4354467b7741385f74683474245f696c6c336147616c2e2e6f32366365666362327d'
```

לאחר מכן תרגמתי לHEX

Paste hex numbers or drop file

```
7069636f4354467b7741385f74683474245f696c6c336147616c2e2e6f3
2366365666362327d
```

Character encoding

ASCII

Convert | Reset | Swap

```
picoCTF{wA8_th4t$_ill3aGal..o26cefcb2}
```

תרגמתי בHEX TO ASCII ומצאתי את הדגל

<u>מה למדתי</u>

למדתי המון מהאתגר הזה לגבי איך עובדת הצפנת RSA, מידע זה יהיה יעיל מאוד לאתגרים הבאים!