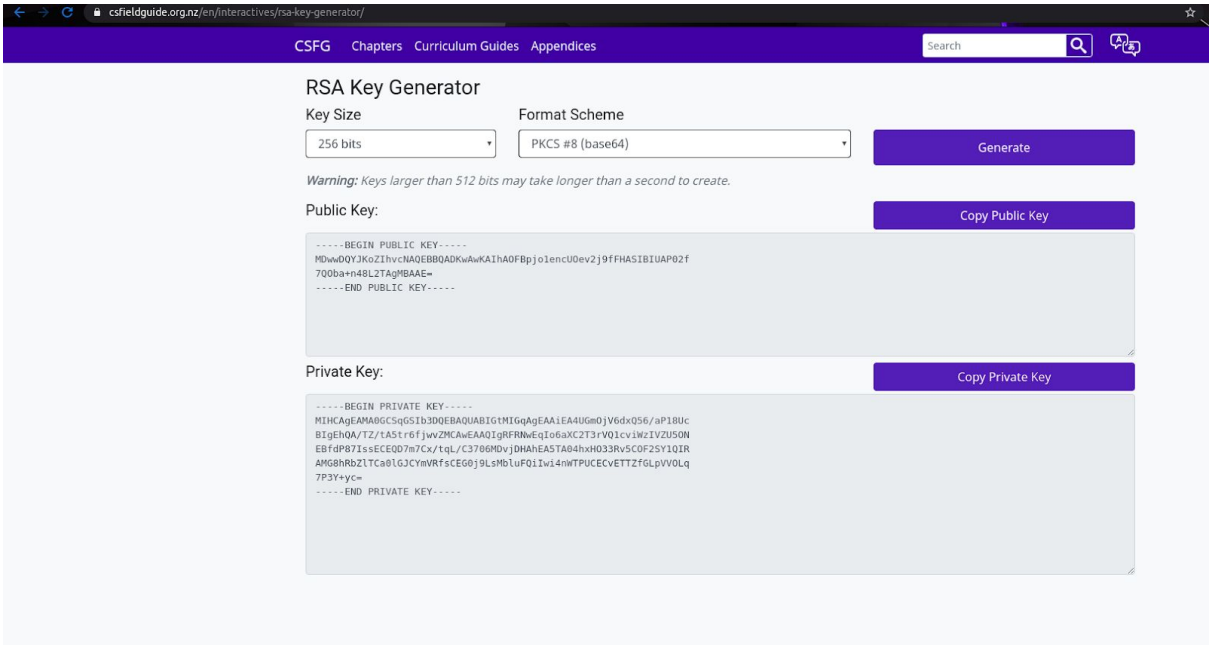
محمد عبدالهی 9530153

-1

از طریق سایت زیر یک کلید 256 بیتی تولید کردیم:



از طریق انجام مراحل زیر یک پیام را توسط کلید عمومی رمز کردیم.

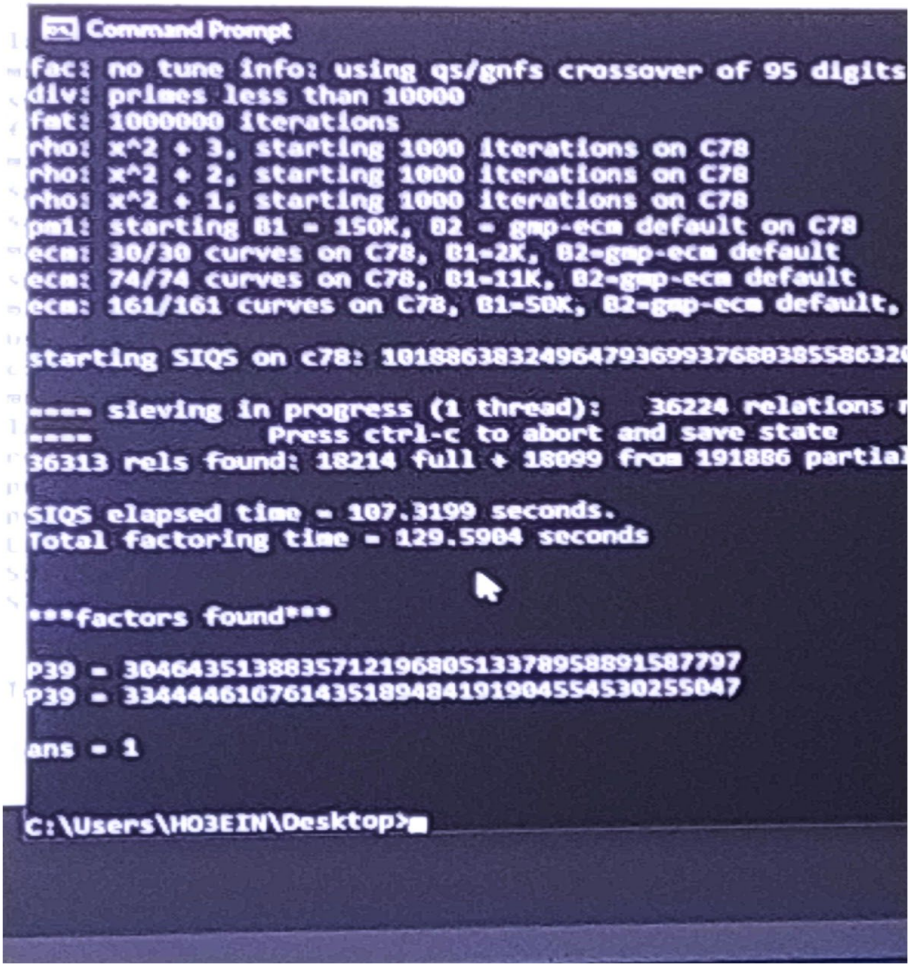
در مرحله بعد لازم بود تا n را محاسبه کنیم.

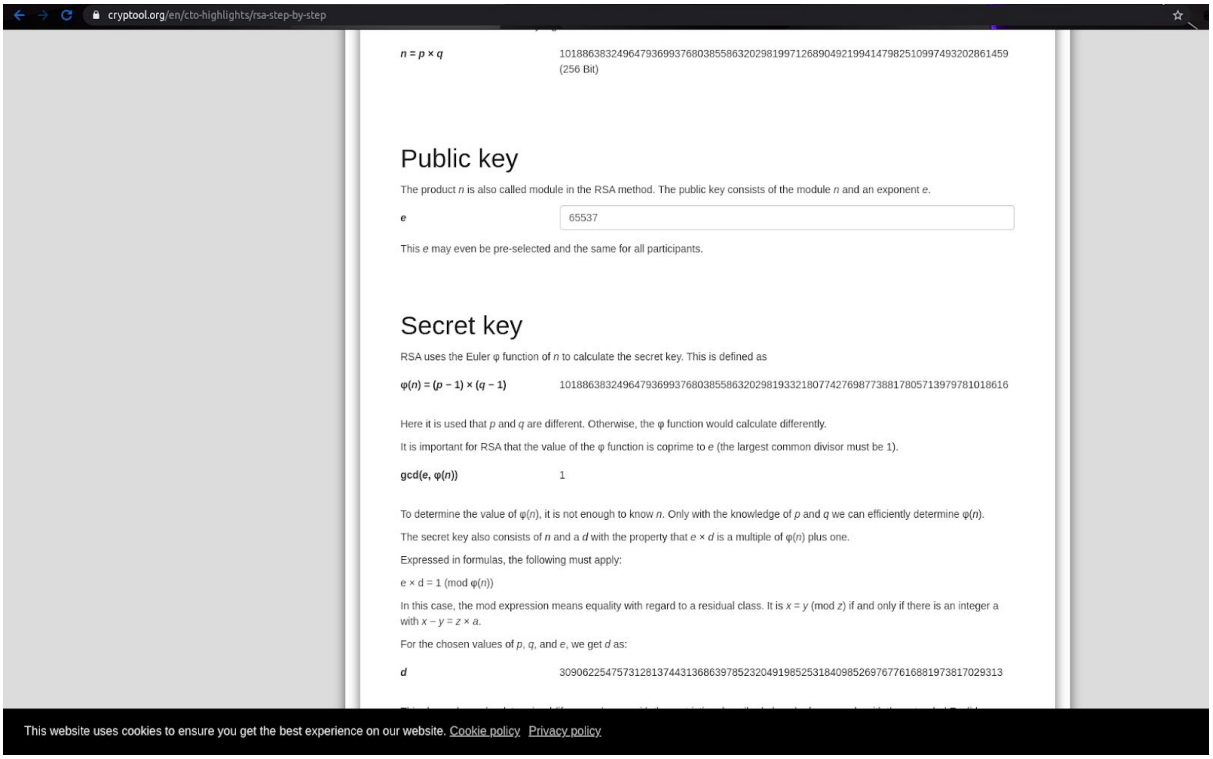
$$n = 101886383249647936993768038558632029819971268904921994147982510997493202861459$$

برای بدست اوردن n مراحل زیر را طی کردیم:

```
mohammad@mohammad-X556UQ ~/Documents/amniat/az/hw2/q1 openssl rsa -pubin -inform PEM -text -noout < public.pem
RSA Public-Key: (256 bit)
Modulus:
    00:e1:41:a6:3a:35:7a:77:14:39:eb:f6:8f:d7:c5:
    1c:04:88:04:85:00:3f:4d:9f:ed:03:9b:6b:e9:f8:
    f0:bd:93
Exponent: 65537 (0x10001)
mohammad@mohammad-X556UQ ~/Documents/amniat/az/hw2/q1 python
Python 2.7.18rc1 (default, Apr  7 2020, 12:05:55)
[GCC 9.3.0] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> s = '00:e1:41:a6:3a:35:7a:77:14:39:eb:f6:8f:d7:c5:1c:04:88:04:85:00:3f:4d:9f:ed:03:9b:6b:e9:f8:f0:bd:93'
>>> s.replace(':')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: replace() takes at least 2 arguments (1 given)
>>> s.replace(':', '')
'00e141a63a357a771439ebf68fd7c51c04880485003f4d9fed039b6be9f8f0bd93'
>>> s = s.replace(':', '')
>>> s.upper()
'00E141A63A357A771439EBF68FD7C51C04880485003F4D9FED039B6BE9F8F0BD93'
>>>
KeyboardInterrupt
>>>
KeyboardInterrupt
>>>
[1]  + 57991 suspended  python
mohammad@mohammad-X556UQ ~/Documents/amniat/az/hw2/q1 bc -q
ibase=16
00E141A63A357A771439EBF68FD7C51C04880485003F4D9FED039B6BE9F8F0BD93
10188638324964793699376803855863202981997126890492199414798251099749\
3202861459
```

سپس از طریق yafu عدد به دست آمده را فکتور کردیم و نتایج حاصل در فایل factor.log ذخیره شده. سپس از طریق دو فاکتور به دست آمده به private key رسیدیم.





و در مرحله اخر کلید را به فرمت مد نظر تبدیل کرده و در فایل private-key.pem ذخیره کرده و با استفاده از کدهای زیر پیام را رمزگشایی کردیم.

قطعا با افزایش طول کلید زمان لازم برای بایپس کردن آن و فکتور کردن چندین برابر خواهد شد.

سوال دوم

برای این منظور از کتابخانه pycryptodome استفاده کردیم. ابتدا کتابخانه را نصب کردیم. سپس در مرحله اول یک کلید عمومی از طریق این کتابخانه برای سرور ایجاد کرده و سپس کلاینت که ارتباط برقرار میکند کلید برای او

ارسال میشود.

```
#generate public key for server
random_generator = Random.new().read
RsaKey = RSA.generate(1024, random_generator)
print('Key generated')
serverPublicKey = RsaKey.publickey().n

#send public key to client
sending_data = input("Press Enter To Send PublicKey For Client")
conn.send(bytes(str(serverPublicKey).encode()))
```

سپس کلاینت این کلید را دریافت کرده و یک کلید خصوصی تولید کرده و با این کلید رمز کرده و برای ما ارسال میکند.

```
# receive server's public key
data = ''
while 1:
    try:
        data = s.recv(1024)
        data = int(data.decode())
        break
    except ConnectionResetError:
        print('Broken PIPE!')


#send aes key for server
serverPublicKey = RSA.construct((data, 65537))
encryptor = PKCS1_OAEP.new(serverPublicKey)
sending_data = str(input("Press Enter To Send SessionKey For Client"))
sessionKey = os.urandom(64)
sending_data = encryptor.encrypt(sessionKey)
s.send(sending_data)
```

در مرحله بعد در سرور این کلید دریافت شده و رمز گشایی میشود.

```
#receive client session key
data = ''
while 1:
    try:
        data = (conn.recv(1024))
        break
    except ConnectionResetError:
        print('Broken PIPE!')

encryptor = PKCS1_OAEP.new(RsaKey)
key = encryptor.decrypt(data)
```

و ازین به بعد پیامها بین کلاینت و سرور با این کلید رد و بدل میشود.

سوال سوم

ابتدا لیستی از دانشجوها با نام های 1 تا 149 و رمز هایی مشخص درست کردیم.

در مرحله بعد با دستور

$$openssl\ genrsa\ -out\ key.pem\ 2048$$

یک کلید به طول 2048 تولید کردیم سپس به رمز کردن نام های موجود در فایل data.json پرداختیم و در فایل encrypted.json ذخیره کردیم. سپس کد مربوط به سرچ را نوشتیم که ابتدا به دنبال نام دانشجو گشته و سپس در صورت یافتن آن شماره دانشجویی مربوطه را رمزگشایی می کند.

خطر احتمالی میتواند در صورتی باشد که نام دو دانشجو یکی باشد در این صورت در خروجی کد دو شماره دانشجویی خواهیم داشت.