

[Features](#) [Business](#) [Explore](#) [Pricing](#)

This repository Search

[Sign in](#) or [Sign up](#)

You must be signed in to star a repository

[nbrisset](#) / [CTF](#)[Watch](#) 1 [★ Star](#) 0 [Fork](#) 1[Code](#) [Issues](#) 0 [Pull requests](#) 0 [Projects](#) 0 [Pulse](#) [Graphs](#)Branch: master ▾ [CTF](#) / [abctf-2016](#) / [challenges](#) / [old-rsa-70](#) /[Create new file](#) [Find file](#) [History](#)

nbrisset new folder Challenges

Latest commit 4f10a01 on 8 Feb

..		
README.md	new folder Challenges	2 months ago
solve_rsa.py	new folder Challenges	2 months ago

[README.md](#)[<<< Return to ABCTF 2016 tasks and writeups](#)

Old RSA (Cryptography, 70 points)

I recovered an RSA encrypted message from the 1980's. can you decrypt it?

c = 29846947519214575162497413725060412546119233216851184246267357770082463030225

n = 70736025239265239976315088690174594021646654881626421461009089480870633400973

e = 3

This is a basic "RSA challenge". We are given an RSA public key (n, e) and an encrypted message (c), which is likely to be the flag. The security of [the RSA algorithm](#) is based on the mathematical difficulty of finding two prime factors of a very large number. However, this challenge has a small modulus (only 77 digits), so it is very easy to recover the two factors p and q (thanks to [factorDB](#) for example), and then easy to decrypt the flag.

Running this Python script gave us the flag and 70 points.

```
import gmpy
from Crypto.PublicKey import RSA

#problem statement
n = 70736025239265239976315088690174594021646654881626421461009089480870633400973
c = 29846947519214575162497413725060412546119233216851184246267357770082463030225
e = long(3)

#thanks to factorDB
p = 238324208831434331628131715304428889871
q = 296805874594538235115008173244022912163

#computing the private key exponent d
d = long(gmpy.invert(e, (p-1)*(q-1)))

#recovering the private key
key = RSA.construct((n,e,d))

print key.decrypt(c)
#6872557977505747778161182217242712228364873860070580111494526546045

print hex(key.decrypt(c))
#0x41424354467b746831735f7761735f683472645f696e5f313938307dL

print hex(key.decrypt(c))[2:-1].decode('hex')
#ABCTF{th1s_was_h4rd_in_1980}
```

