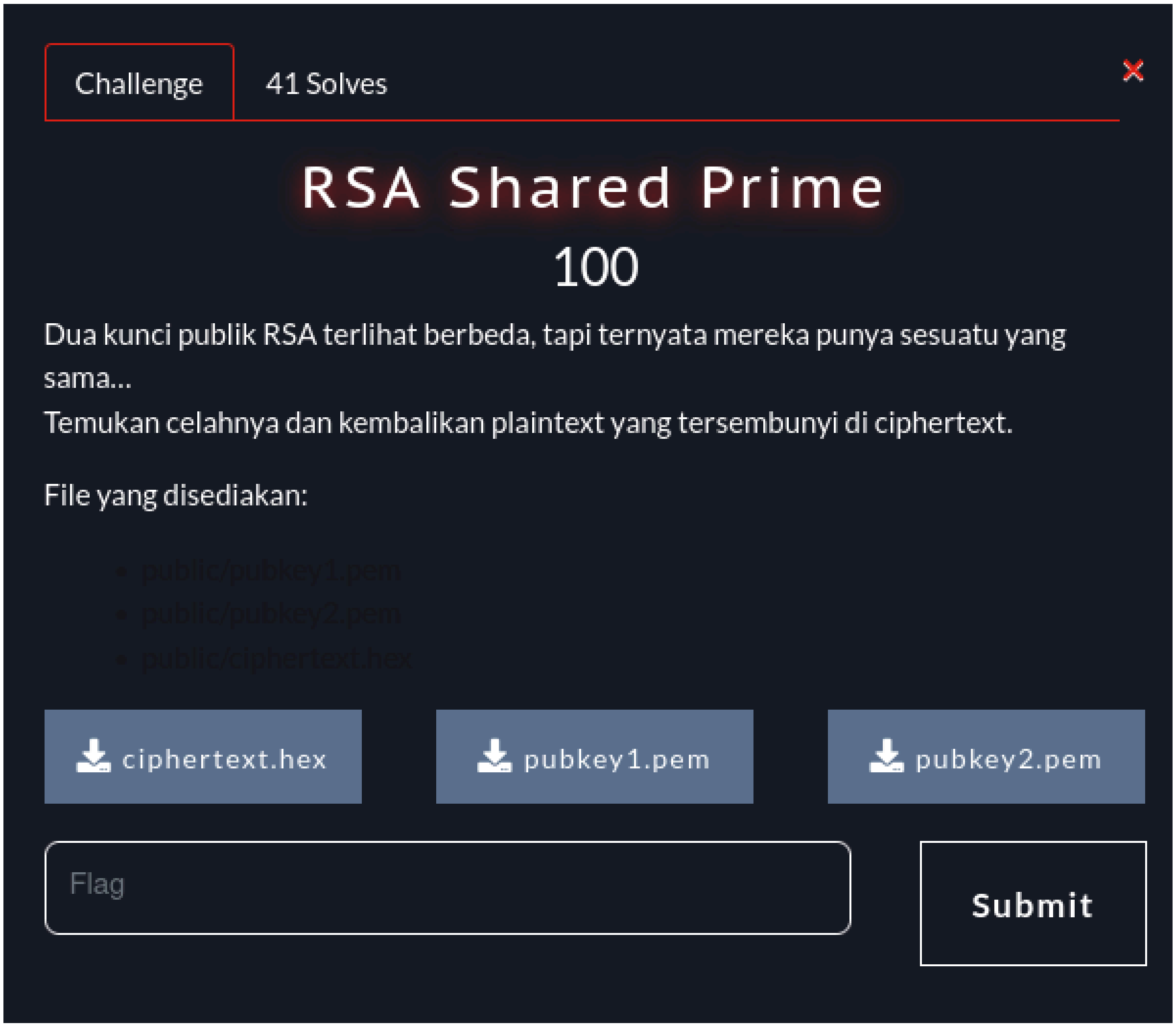
Write-up RSA Shared Prime

Player: Constantine
Kategori: Cryptography



RSA Shared Prime

100

Dua kunci publik RSA terlihat berbeda, tapi ternyata mereka punya sesuatu yang sama...
Temukan celahnya dan kembalikan plaintext yang tersembunyi di ciphertext.

File yang disediakan:

- public/pubkey1.pem
- public/pubkey2.pem
- public/ciphertext.hex

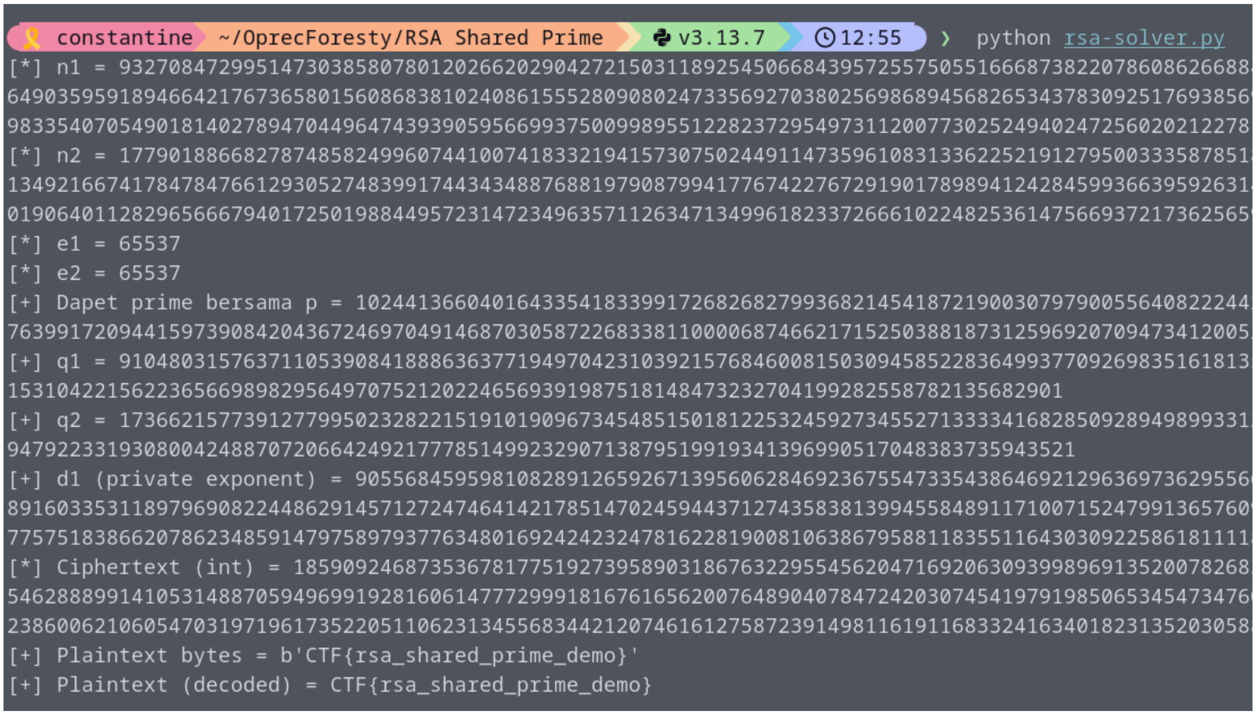
Langsung Ambil Flag

Aku langsung lempar ketiga file ini ke ChatGPT untuk buatkan script python rsa-resolver.py. Dari situ langsung dibuatkan script **rsa-resolver.py**:

```
from Crypto.PublicKey import RSA
from Crypto.Util.number import long_to_bytes
from math import gcd

# 1. Import 2 public key RSA
with open("pubkey1.pem", "rb") as f:
    k1 = RSA.import_key(f.read())

with open("pubkey2.pem", "rb") as f:
    k2 = RSA.import_key(f.read())

n1, e1 = k1.n, k1.e
n2, e2 = k2.n, k2.e

print("[*] n1 =", n1)
print("[*] n2 =", n2)
print("[*] e1 =", e1)
print("[*] e2 =", e2)

# 2. Cari prime yang dishare pakai gcd
p = gcd(n1, n2)

if p == 1:
    raise SystemExit("[-] Tidak ada prime yang dishare (gcd = 1). CTF ini seharusnya shared prime, cek lagi filenya.")

print("[+] Dapet prime bersama p =", p)

# 3. Faktorkan kedua modulus
q1 = n1 // p
q2 = n2 // p

print("[+] q1 =", q1)
print("[+] q2 =", q2)

# Sanity check
assert p * q1 == n1
assert p * q2 == n2

# 4. Bangun private key untuk kunci pertama (misalnya ciphertext pakai pubkey1)
phi1 = (p - 1) * (q1 - 1)
```

```
# Python 3 punya modular inverse di pow
d1 = pow(e1, -1, phi1)

print("[+] d1 (private exponent) =", d1)

# 5. Baca ciphertext (format hex)
with open("ciphertext.hex", "r") as f:
    c_hex = f.read().strip()

c = int(c_hex, 16)
print("[*] Ciphertext (int) =", c)

# 6. Decrypt
m = pow(c, d1, n1)
pt = long_to_bytes(m)

print("[+] Plaintext bytes =", pt)
try:
    print("[+] Plaintext (decoded) =", pt.decode())
except UnicodeDecodeError:
    print("[+] Plaintext bukan plain UTF-8, cek bentuk lain (mungkin base64
/ flag format khusus).")
```

Screenshot output decrypt:



```
 constantine  ~/OprecForesty/RSA Shared Prime  v3.13.7  12:55  ❯ python rsa-solver.py
[*] n1 = 93270847299514730385807801202662029042721503118925450668439572557505516668738220786086266688
64903595918946642176736580156086838102408615552809080247335692703802569868945682653437830925176938569
98335407054901814027894704496474393905956699375009989551228237295497311200773025249402472560202122 78
[*] n2 = 177901886682787485824996074410074183321941573075024491147359610831336225219127950033358785 1
134921667417847847661293052748399174434348876881979087994177674227672919017898941242845993663959263 1
01906401128296566679401725019884495723147234963571126347134996182337266610224825361475669372173625659
[*] e1 = 65537
[*] e2 = 65537
[+] Dapet prime bersama p = 102441366040164335418339917268268279936821454187219003079790055640822244
763991720944159739084204367246970491468703058722683381100006874662171525038818731259692070947341200 5
[+] q1 = 910480315763711053908418886363771949704231039215768460081503094585228364993770926983516181 3
153104221562236566989829564970752120224656939198751814847323270419928255878213568 2901
[+] q2 = 173662157739127799502328221519101909673454851501812253245927345527133334168285092894989933 1
94792233193080042488707206642492177785149923290713879519919341396990517048383735943521
[+] d1 (private exponent) = 905568459598108289126592671395606284692367554733543864692129636973629556
891603353118979690822448629145712724746414217851470245944371274358381399455848911710071524799136576 0
77575183866207862348591479758979377634801692424232478162281900810638679588118355116430309225861811 11
[*] Ciphertext (int) = 185909246873536781775192739589031867632295545620471692063093998969135200782 68
546288899141053148870594969919281606147772999181676165620076489040784724203074541979198506534547347 6
238600621060547031971961735220511062313455683442120746161275872391498116191168332416340182313520305 8
[+] Plaintext bytes = b'CTF{rsa_shared_prime_demo}'
[+] Plaintext (decoded) = CTF{rsa_shared_prime_demo}
```

Hasil plaintext:

CTF{rsa_shared_prime_demo}

Final Flag

FORESTY{rsa_shared_prime_demo}