Cross Check

CTF: ASIS 2015 (/challenges/ctf/ASIS%202015/)
Category: Cryptography (/challenges/category/Cryptography/)
Write-up *by **GSAir** on May 12, 2015, 5:22 a.m.*

Cross Check

Challenge Description

The flag is encrypted by this code, can you decrypt it?

Material

1. A python script
2. A text file with lot of data

Solution

- The flag has been split in three parts and encrypted with 3 different RSA keys. Hum it should make things much more complex... Oh wait no, there are first generated two prime numbers p1 and q1 and then take the next two primes for both: p2, p3 and q2, q3.
- Using that we compute `N = N1 * N2` , and try to factorize it with Fermat method. This method is very efficient if `N = a * b` with `a` very close form `b` .
- In our case we have `N = N1 * N2 = p1 * q3 * p2 * q2` , so there are multiple two close factors: `(p1 * q3, p2 * q2)` , `(p1 * q2, p2 * q3)` . Thanks to the method they used to generate the prime, `p1` is close to `p3` and `q2` is close to `q3` , `p1 * q2` is closed to `q3 * p2` !!!
- The second factorization is very interesting, if we manage to find `a = p1 * q2` , then `p1 = gcd(N1, a)` and `q2 = gcd(N2, a)` and in addition `b = p2 * q3` . Then we can do the same for `N3` using `N = N2 * N3`

Here is my complete code: (I split the all.txt file in multiple files):

Note: the variable name are `px qx` for `Nx` .

```
import gmpy
import random
from Crypto.PublicKey import RSA
from Crypto.Cipher import PKCS1_v1_5
from Crypto.Cipher import PKCS1_OAEP
import base64
from fractions import gcd

with open("N1.txt") as f:
    N1 = eval(f.read())

with open("N2.txt") as f:
    N2 = eval(f.read())

with open("N3.txt") as f:
    N3 = eval(f.read())

def fermat_factor(n, n1, n2):
    a = gmpy.sqrt(n) + 1
    b = a*a - n
    bsq = gmpy.sqrt(b)

    while bsq * bsq != b or a + bsq == n1 or a + bsq == n2:
        a += 1
        b = a * a - n
        bsq = gmpy.sqrt(b)

    return  a + bsq

N = N1 * N2
a = fermat_factor(N, N1, N2)

p1 =  gcd(N1, a)
q1 =  N1 / p1
assert(N1 == q1 * p1)

q2 = gcd(N2, a)
p2 = N2 / q2
assert(N2 == q2 * p2)

N = N2 * N3
a = fermat_factor(N, N2, N3)

q3 = gcd(N3, a)
p3 = N3 / q3
assert(N3 == q3 * p3)

with open("flag1.txt") as f:
    f1 = f.read()
with open("flag2.txt") as f:
```

```
    f2 = f.read()
with open("flag3.txt") as f:
    f3 = f.read()

f1 = base64.b64decode(f1)
f2 = base64.b64decode(f2)
f3 = base64.b64decode(f3)

e = 65537
d1 = gmpy.invert(e, (p1 -1) * (q1 - 1))
RSA1 = RSA.construct((long(N1), long(65537), long(d1), long(p1), long(q1)))
key1 = PKCS1_v1_5.new(RSA1)

d2 = gmpy.invert(e, (p2 -1) * (q2 - 1))
RSA2 = RSA.construct((long(N2), long(65537), long(d2), long(p2), long(q2)))
key2 = PKCS1_v1_5.new(RSA2)

d3 = gmpy.invert(e, (p3 -1) * (q3 - 1))
RSA3 = RSA.construct((long(N3), long(65537), long(d3), long(p3), long(q3)))
key3 = PKCS1_v1_5.new(RSA3)

print key1.decrypt(f1, None) + key2.decrypt(f2, None) + key3.decrypt(f3, None)
```

Flag: ASIS{a0c8f997d5cdd699d336b0f2f12af326}
