

QIWI CTF 2016 - Crypto 400_1

Nov 18, 2016 • By [thezero](#)

Category: [writeups](#)

Tags: [crypto](#) [qiwictf-2016](#)

Crypto 400_1

Crypto - 400 Points

Message m has been encrypted by RSA with exponent $e=3$ for three users. Users have been used different modulus (n_1 , n_2 , n_3 respectively). As a result 3 ciphertexts have been obtained (c_1 , c_2 , c_3 respectively). Decrypt the message. The flag is a sensible text.

The given **n_1** , **n_2** , **n_3** , **c_1** , **c_2** , **c_3** are reported in the script

Writeup

This crypto challenge is based on the [Håstad's broadcast attack](#).

So by implementing the Chinese Remainder Theorem we could solve this easily

The python script FTW

```
#!/usr/bin/env python

e=3
n1=95118357989037539883272168746004652872958890562445814301889866630723524217
n2=983641659192512462438466673235423180228042348336779241611757332536895813936
n3=688279409393531896130903922268981550217427728978224384835450219442158121468
c1=648304467081690127664145873275688124211304348175260891461901367964612985920
c2=969074907173443465884324916037223126942086603342829642344876876545939847141
c3=436838749130117465300561031454452502813077326340454374865246051046397854690
```

```
def chinese_remainder(n, a):
    sum = 0
    prod = reduce(lambda a, b: a*b, n)
    for n_i, a_i in zip(n, a):
        p = prod / n_i
        sum += a_i * mul_inv(p, n_i) * p
    return sum % prod

def mul_inv(a, b):
    b0 = b
    x0, x1 = 0, 1
    if b == 1: return 1
    while a > 1:
        q = a / b
        a, b = b, a%b
        x0, x1 = x1 - q * x0, x0
    if x1 < 0: x1 += b0
    return x1


def find_invpow(x, n):
    """Finds the integer component of the n'th root of x,
    an integer such that y ** n <= x < (y + 1) ** n.
    """
    high = 1
    while high ** n < x:
        high *= 2
    low = high/2
    while low < high:
        mid = (low + high) // 2
        if low < mid and mid ** n < x:
            low = mid
        elif high > mid and mid ** n > x:
            high = mid
        else:
            return mid
    return mid + 1

flag_cubed = chinese_remainder([n1, n2, n3], [c1, c2, c3])
flag = find_invpow(flag_cubed, 3)

print "flag: ", hex(flag)[2:-1].decode("hex")
# flag: theoretical_computer_scientist_johan_torkel_hastad
```

0 Comments

P=NP CTF Team

 Login ▾ Recommend Share

Sort by Newest ▾



Start the discussion...

Be the first to comment.

 Subscribe Add Disqus to your site

Add Disqus Add

 Privacy

PequalsNP -|- visit us on [github](#) - [twitter](#) - [cftime.org](#) actually collaborating with [JBZ team](#)
subscribe [via RSS](#)