Cyber Santa is Coming to Town: XMAS Spirit

Author crxzii | Contest Date 05.12.2021

Solve Moment During The Contest | Category Cryptography | Score 300

Description

> Now that elves have taken over Santa has lost so many letters from kids all over the world. However, there is one kid who managed to locate Santa and sent him a letter. It seems like the XMAS spirit is so strong within this kid. He was so smart that thought of encrypting the letter in case elves captured it. Unfortunately, Santa has no idea about cryptography. Can you help him read the letter?

Attached Files

- challenge.py

```
import random
from math import gcd

def encrypt(dt):
    mod = 256
    while True:
        a = random.randint(1,mod)
        if gcd(a, mod) == 1: break
    b = random.randint(1,mod)

    res = b''
    for byte in dt:
        enc = (a*byte + b) % mod
        res += bytes([enc])
    return res

dt = open('letter.pdf', 'rb').read()

res = encrypt(dt)

f = open('encrypted.bin', 'wb')
f.write(res)
f.close()
```

- encrypted.bin

The encrypted PDF file

Summary

We know that our original file was a PDF file. So our plan is to brute force our a and b by checking if '%PDF' (the standard PDF header) is included in the first few bytes of our decryption.

Once we got a and b, we can decrypt the whole file.

Flag

```
HTB{4ff1n3_c1ph3r_15_51mpl3_m47h5}
```

Detailed Solution

First of all, we don't need to go through all 256 values for a, because we know, that `gcd(a, 256) = 1`.
We'll write a function to give us all possibilities for a:

```
def get_possible_a():
    mod = 256
    arr = []
    for a in range(1,257):
        if gcd(a, mod) == 1:
            arr.append(a)

    return arr
```

We want to then extract the first few bytes in the `encrypted.bin` and decrypt it. We write a function, that decrypts the bytes:

```
def decrypt(msg, a, b):
    res = b''
    for char in msg:
        char = char - b
        char = a * char % 256
        res += bytes([char])
    return res
```

We then want to write a function, that decrypts the first few bytes and checks if the decryption includes '%PDF'. If it does, we got our a and b:

```
def find_pdf_string(msg):
    for a in get_possible_a():
        for b in range(1,257):
            start = decrypt(msg, a, b)
            if b'%PDF' in start:
                print("[+] Found a: {} and b: {}".format(a, b))
                return a,b
```

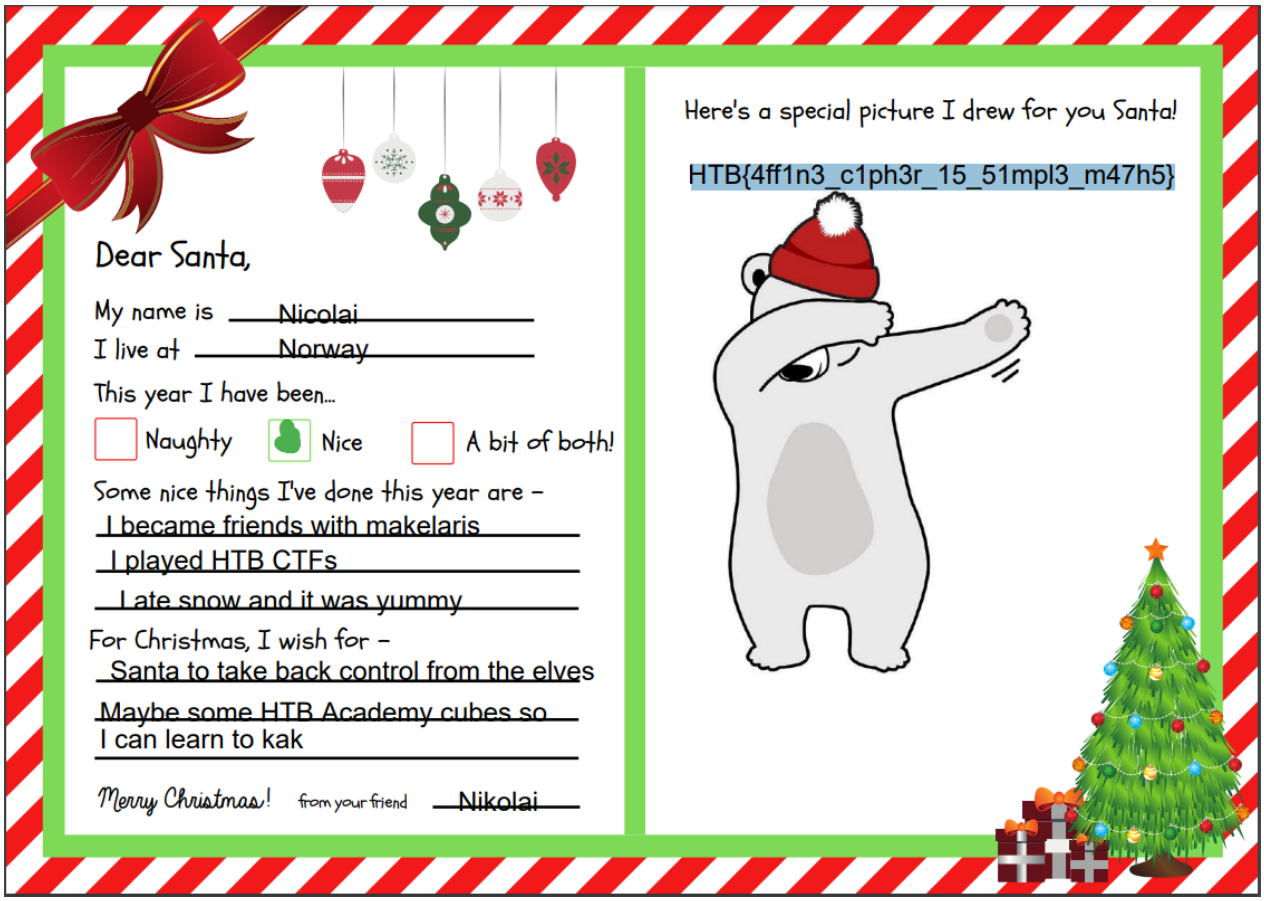
Once we got our a and b, we can just pass all of the encrypted bytes into our decrypt function and save the decryption in a pdf file.

When we combine all of that, we get a python program, that solves everything automatically for us:

```
from math import gcd

def get_possible_a():
    mod = 256
    arr = []

    for a in range(1,257):
        if gcd(a, mod) == 1:
            arr.append(a)

    return arr
```

```python
def decrypt(msg, a, b):
    res = b''
    for char in msg:
        char = char - b
        char = a * char % 256
        res += bytes([char])
    return res

def find_pdf_string(msg):
    for a in get_possible_a():
        for b in range(1,257):
            start = decrypt(msg, a, b)
            if b'%PDF' in start:
                print("[+] Found a: {} and b: {}".format(a, b))
                return a,b

dt = open('encrypted.bin', 'rb').read()
a,b = find_pdf_string(dt[:5])

res = decrypt(dt,a,b)
f = open('dec.pdf', 'wb')
f.write(res)
f.close()

print("[+] Decrypted PDF file to 'dec.pdf'!")
```

Our PDF looks like this:



There we can easily spot our flag: `HTB{4ff1n3_c1ph3r_15_51mpl3_m47h5}`