A Curious Case of a Double RSA Encryption

Hafizh Afkar Makmur - 13514062[1]
Informatics/Computer Science Major
School of Electrical Engineering and Informatics
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
[1]*13514062@std.stei.itb.ac.id*

***Abstract*—This paper introduces an interesting problem asked in Tokyo Westerns/MMA CTF 2nd 2016 and its problem. The problem contains an encrypted file and a source code of the encryption program. The encryption program contains double RSA encryption so it produces two kinds of public key. Here it is proven that it is a blunder to do the double encryption and it is to break the private key using the two given public key. The solution proves the writer's gained skill in both information security especially cryptography and programming language Python.**

***Keywords*—RSA, python, CTF, information security**

I. Introduction

Nowadays, information are everywhere and someone can get an information on certain things easily. Obviously, it is not always good to give a person's information to everyone because it can easily fall to irresponsible people who can easily use it to do fraud and et cetera. That's why someone needs to protect their privacy and use certain knowledge known as information security to better protect their information from illegal access. One of the means of information security is to do encryption to some information to make it unreadable to anyone but the owner of the document.

One of the methods of encryption still used an the moment is known as Rivest-Shamir-Adleman encryption or popularized as RSA encryption. The encryption [1] is an asymmetrical encryption which uses public key to encrypt something and private key to decrypt it. This method hinges on the difficulty of factorizing prime factors of a number. This method is tested and proven even until now, 40 years after its invention.

Capture-The-Flag of CTF is a contest for people to prove their prowess on information security knowledge. The contest is useful for anyone who wants to learn more about information security and wants to test their knowledge against others. One of the most famous CTF in the world is DEFCON which is started is 1992 and held regularly every year until now. Various similar contest are held in many universities to test and teach their own students and friends about information security.

At 3 September 2016 until 5 September 2016 Tokyo Westerns and MMA held a CTF contest [2]. The contest was held online and challenges various genres from Cryptography, Programming, Reversing, Forensics, to Web knowledge. The contest was in Jeopardy format. Which means some questions are published and each time must compete to solve the questions as much and as fast as possible. The official link of the contest is https://tokyowesterns.github.io/ctf2016/ and now everyone can see the questions and the solution of each question for further study.

The writer competed in this contest under username *codebender* and met and solve one particular cryptography problem about RSA. The writer declares that he solves this problem on his own as can be seen under problem solver section on the official website. The problem is "Twin Prime" which is a Crypto and Warmup problem weighted 50 points.

II. Problem Statement

A. Problem Structure

The problem gives contestant 4 files: encrypt.py, encrypted, flag1, and flag2. Apparently, there was one file named flag which contains flag or answer for this problem. But now the flag is encrypted in the file named encrypted and now the contestant must figure out what is the content of the flag file to figure out the answer to this question.

B. Encryption Program

It is known then that file flag in encrypted with a Python script in encrypt.py. The file generates file encypted, flag1, and flag2. Encrypt.py uses RSA encryption in its encryption method and then generates two public key in file flag1 and flag2. Here is attached the encryption method in file encrypt.py.

```
def getTwinPrime(N):
    while True:
        p = getPrime(N)
        if isPrime(p+2):
            return p

def genkey(N = 1024):
    p = getTwinPrime(N)
    q = getTwinPrime(N)
    n1 = p*q
    n2 = (p+2)*(q+2)
    e = long(65537)
    d1 = inverse(e, (p-1)*(q-1))
    d2 = inverse(e, (p+1)*(q+1))
    key1 = RSA.construct((n1, e, d1))
    key2 = RSA.construct((n2, e, d2))
```

```
    if n1 < n2:
        return (key1, key2)
    else:
        return (key2, key1)

rsa1, rsa2 = genkey(N)
```

As is seen, the method searches for twin primes p and q and use it for double RSA encryption. The program then generates two public key known as n1 and n2 here and make two RSA key to make the encryption feasible. The idea is one must has the two public key plus e variable to make an encrypted document and therefore can send it securely to the owner of the private key. Just like RSA, the program is supposedly secure because there are difficulties to retrieve p and q from n1 and n2 so the document is secure even if the attacker knows the value of n1, n2, and e. It is even seemingly more secure because the document is encrypted twice with an already strong encryption. But is it really more secure than a normal RSA encryption?

Nope.

III. PROBLEM SOLUTION BREAKDOWN ANALYSIS

A. Insight for solving the problem

First we must realize that two public key that we have, n1 and n2, is related. One equals p*q and the other one equals (p+2)*(q+2). Then, one must know that actually does not need to know the value of p and q exactly. To break RSA, one only needs to know the value of (p-1)*(q-1) or in this case, both (p-1)*(q-1) and (p+1)*(q+1). Normal RSA is hard because we only know one number (n) to retrieve two number (p and q) to solve for (p-1)*(q-1). But know we have two number (n1 and n2) to solve four number (p, q, p+2, q+2) and ultimately get two numbers ((p-1)*(q-1) and (p+1)*(q+1)). But because n1 and n2 is supposedly related, the breakdown become easier and therefore possible.

B. Mathematical analysis

Our target is both (p-1)*(q-1) and (p+1)*(q+1). They are all both equal to

$$(p-1)*(q-1) = pq - (p+q) + 1$$

and

$$(p+1)*(q+1) = pq + (p+q) + 1$$

respectively. To get both value, we only need to know the value of pq and p+q and plug it into the equation to get the solution.

We already have two known value n1 and n2 which equal to

$$n1 = p*q$$

and

$$n2 = (p+2)*(q+2) = pq + 2(p+q) + 4.$$

With this known value, we already got one of the value that we'd like to know pq which like we have already seen is equal to n1. Now how do we get p+q? Look carefully in n2 that there is the value of p+q there. But how do we retrieve that value? The only other unknown value in n2 equation is pq which we can already got from n1. Therefore, we can retrieve p+q from n2 using n1 in the following equation

$$n2 - n1 = pq + 2(p+q) + 4 - pq = 2(p+q) + 4$$

$$2(p+q) = n2 - n1 - 4$$

$$p+q = \frac{n2 - n1 - 4}{2}$$

Voila! Now we know both the value of pq and p+q. Therefore, we know the value of (p-1)*(q-1) and (p+1)*(q+1) which is

$$\begin{aligned}(p-1)*(q-1) &= pq - (p+q) + 1 \\ &= n1 - \frac{n2-n1-4}{2} + 1 \\ &= \frac{3*n1-n2+6}{2}\end{aligned}$$

and

$$\begin{aligned}(p+1)*(q+1) &= pq + (p+q) + 1 \\ &= n1 + \frac{n2-n1-4}{2} + 1 \\ &= \frac{n1+n2-2}{2}\end{aligned}$$

respectively. Knowing both the value of (p-1)*(q-1) and (p+1)*(q+1), we can know devise the counter to the encryption program encrypt.py.

C. Implementaion of the Solution

To create a counter to the encryption program encrypt.py, we must make a function which can generate exactly like genkey function as attached above, but with different input to include all value that we know just like n1, n2, and e. Therefore, we can devise a function like this

```
def genkey(n1,n2,e):
    d1 = inverse(e, (3*n1-n2+6)/2)
    d2 = inverse(e, (n1+n2-2)/2)
    key1 = RSA.construct((n1, e, d1))
    key2 = RSA.construct((n2, e, d2))
    if n1 < n2:
        return (key1, key2)
    else:
        return (key2, key1)

rsa1, rsa2 = genkey(N)
```

We can see that the function is almost exactly the same as the attached function but with both (p-1)*(q-1) and (p+1)*(q+1) changed to be computed from n1, n2, and e. With this, we can get the two RSA keys used for double RSA encryption and decrypt the file encrypted twice first using rsa2 and then rsa1, reversing the process. After that, we can get the flag and the problem is solved.

IV. Lesson Learned and Conclusion

One of the most famous quotes from Bruce Scheneier [3], a prominent figure in Cryptography, is that everyone, be him an amateur or a very excellent mathematician which often devise various cryptography methods, can make a cryptography method that he can’t break. It is known as Schneier’s Law. It means that it is not enough knowing that you can’t break your cryptography program to ensure that your program is secure. The only known test that is effective to test a cryptography program is time. A cryptography method is known as secure if it can withstand various method of various people trying to break it and failed. It is proven here that we see a long standing method known as RSA, a long known encryption method, can be broken by an attempt to “reinforce” it, supposedly by doubling it and tweaking it a little bit. Some modification or even a construction of a new encryption algorithm must be handled carefully and tested frequently not by the author himself.

V. Acknowledgment

The first and foremost thanks from the author is to the God, the All-Seeing, All-Knowing, for giving the author strength and time to finish this task completely and on time. Praise be upon Him.

Second thanks is for the writer’s teachers Mr. Rinaldi .M, Mrs. Ayu .P, and Mrs. Deasy to give the writer an opportunity to write this paper to prove the writer’s prowess in information security especially cryptography and Python programming language. May you all be full of luck in everything you do.

Another thanks for all my friends who give me inspiration and support for studying various subjects I’ll never know about without them. May a wonderful present and future be with them.

Final thanks for the host of Tokyo Westerns/MMA CTF 2nd 2016 esepcially ytoku and nomeaning who write this beautifiul problem “Twin Primes” for the author to solve. May successful career and excellence be blessed upon them.

References

[1] R. Rivest, A. Shamir and L. Adleman, "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems," *Communications of the ACM,* vol. 2, no. 21, pp. 120-126, 1978.

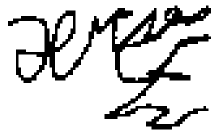
[2] "Tokyo Westerns/MMA CTF 2nd 2016," Tokyo Westerns & MMA, 5 9 2016. [Online]. Available: https://tokyowesterns.github.io/ctf2016/. [Accessed 5 5 2017].

[3] B. Schenier, "Memo to the Amateur Cipher Designer," 15 10 1998. [Online]. Available: http://www.schneier.com/crypto-gram-9810.html#cipherdesign.

Statement

With this statement I declare that theis paper that I write is my own writing, not an adaptation, nor a translation of other person’s paper, and not a plagiarism.

Bandung, 5 Mei 207



Hafizh Afkar Makmur - 13514062