A Comparison of methods for attacking RSA

Yichen Zhou
yichenzhou@umail.ucsb.edu
Department of Computer Science
University of California, Santa Barbara

June 13, 2018

Abstract

This paper examines how to implement two different attacks on the public-key cryptosystem RSA and analyze their performance. It includes Fermat's factorization and Wiener's attack. Both algorithms are implemented in Python and tested with different parameters for RSA system.

1 Introduction

Rivest-Shamir-Adleman, also known as RSA [2], is a widely used public-key cryptosystems. It uses several keys to encrypt and decrypt information. One can encrypt a message to another using the public key pair (n, e). Then the plaintext can be recovered from ciphertext using the private key pair (n, d). The safety of RSA is based on the difficulty of the factorization of the product of two large prime numbers [2]. Many attack methods such as Fermat factoring and Wiener's attack can efficiently find the secret key d.

2 RSA

The RSA is introduced by R.L.Rivest, A.Shamir and L.Adleman. The keys for RSA algorithm are generated in the following way [2]. First choosing two distinct prime numbers p and q and computing $n = pq$. Then computing $\phi(n) = (p-1)(q-1)$. After choosing a integer e, such that $1 < e < \phi(n)$ and e and $\phi(n)$ are relatively prime, we can compute $d = e^{-1}(mod\ \phi(n))$.

Therefore, RSA generates public keys (n, d) and private keys (n, e) based on two large prime numbers. Thus, one can encrypt a message m to another using the public key pair (n, e). The ciphertext c can be computed such that $c = m^e(mod\ n)$. Then the plaintext m can be recovered from c by computing $m = c^d(mod\ n)$ using the private key pair (n, d).

3 Fermat's Factorization Method

Since the private key d is generated by computing $d = e^{-1}(mod\ \phi(n))$, it can be calculated if $\phi(n)$ is known. $\phi(n)$ is equal to the product of p and q, and Fermat's factorization method can factor n into p and q efficiently. The algorithm is given below.

function FermatFactorization(n)
input: n
output: p, q such that $p \cdot q = n$
1: **if** n is even:
1b: **return** [2,$n/2$]
2: $a \leftarrow \lceil(\sqrt{n})\rceil$
3: $b \leftarrow a \cdot a - n$
4: **while** b is not a square:
4a: $a \leftarrow a + 1$
4b: $b \leftarrow a \cdot a - n$
5: **return** $[a - \sqrt{b}, a + \sqrt{b}]$

Fermat's factorization method is very useful if the gap between p and q is small. In this case, Fermat's factorization method can factor n quickly so that private key d will be exposed. It is unsafe to use two weak primes which are very close to each other.

4 Wiener's attack

It has been proved that the d can be efficiently found using Wiener's attack when the private key d is small [3].

Theorem 1 (Wiener's Theorem [1]) *Let $n = p \cdot q$ with $q < p < 2q$ and $d < \frac{1}{3} \cdot n^{\frac{1}{4}}$. Given (n, e) with $e \cdot d = 1(mod\ \phi(n))$, the attacker can efficiently recover d.*

First of all, a continued fraction is an expression in the following form:

$$x = a_0 + \frac{1}{a_1 + \frac{1}{a_2 + \frac{1}{a_3 + \cdots}}}.$$

Thus, a continued fraction can be abbreviated by its continued fraction expansion such that $x = [a_0, a_1, a_2, a_3, \cdots]$. Moreover, since

$$\begin{aligned} ed &= 1 \pmod{(p-1)(q-1)} \\ ed &= k \cdot (p-1)(q-1) + 1 \\ \frac{e}{pq} &= \frac{k}{d} \cdot (1 - \delta),\ where\ \delta = \frac{p + q - 1 - \frac{1}{k}}{pq} \\ \frac{e}{n} &= \frac{k}{d} \cdot (1 - \delta) \end{aligned}$$

So that $\frac{k}{d}$ is slightly bigger than $\frac{e}{n}$. Also because $\phi(n) = (p-1)(q-1) = n-p-q-1 = n-p-\frac{n}{p}+1$, n satisfies the equation $p^2 + p(\phi(n) - n - 1) - n = 0$. By using above theorems and properties, the algorithm description of Wiener's attack is given below.

function WienerAttack(n, e)
input: n, e
output: $\phi(n)$
1: Find all convergents $\frac{k}{d}$ of the continued fraction expansion of $\frac{e}{n}$: $\frac{k_1}{d_1}, \frac{k_2}{d_2}, \frac{k_3}{d_3}, \cdots, \frac{k_m}{d_m}$
2: $i \leftarrow 1$
3: $\phi(n) = \frac{ed_i - 1}{k_i}$
4: **while** $i <= m$ **and** $p^2 + p(\phi(n) - n - 1) - n = 0$ doesn't have two prime roots:
4a: $i \leftarrow i + 1$
4b: $\phi(n) = \frac{ed_i - 1}{k_i}$
5: **if** $i <= m$
5b: **return** $\phi(n)$

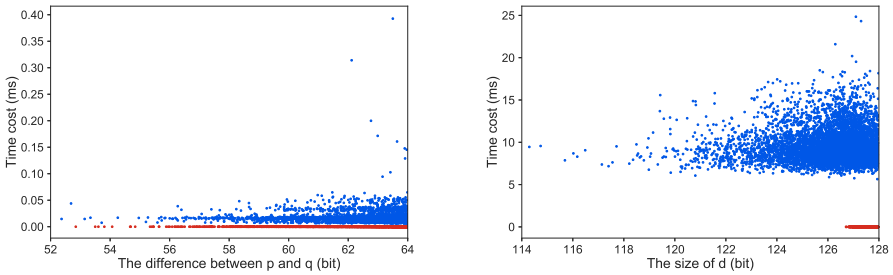
If we obtain p, q and $\phi(n)$ using the above Wiener's attack algorithm, we can calculate the private key d. Thus it is also unsafe when d is small.

5 Implementation and Experiment

I use Python to implement the two algorithms I describe above and run code on my personal computer. I test two attack methods with different RSA keys which are generated by 256-bit p and 256-bit q.

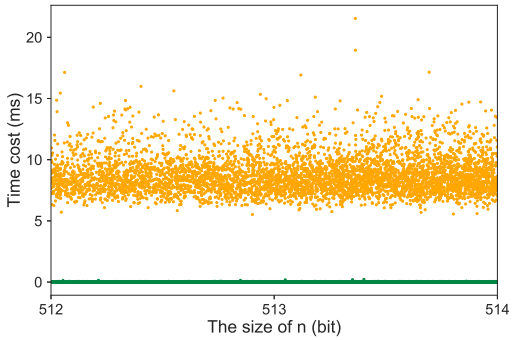
When testing Fermat's factorization method, I want to verify the relation between two variables, the timecost and the difference between p and q. So I randomly choose 256-bit p and q, and I test Fermat's factorization method with different n generated by p and q. I use 10000 random (p, q) pairs to test and the result is shown in Fig. 1(a). The blue dots represent the time cost when the attack succeeds while the red dots represent the attack fails. It shows that the time cost of the successful attack tends to increase when the difference between p and q increases. Also the method is more likely to fail when the difference between p and q becomes bigger.

Similarly, when testing Wiener's attack, I want to see the relationship between two variables, the time cost and d. Thus I randomly choose 566625432812713312026715234631662258855659896587416017147083282136553645395574 and $q = 1054453548201201698218618988165831055334242690006898896539520380032525082796091$. Given $n = p \cdot q$, I test Wiener's attack with 10000 d which is qualified and randomly generated as well. The result is shown in Fig. 1(b). It demonstrates that the time cost and probability of failure both increase with d. Notice that when $d > 13949775742836522512439930739618621140 7$ the attack starts to fail, which accords with Wiener's theorem because $13949775742836522512439930739618621140 7 > \frac{1}{3} \cdot n^{\frac{1}{4}} = 9.26743661559 \cdot 10^{37}$.



(a) Fermat's factorization Method

(b) Wiener's attack



(c) Comparison between Fermat's factorization Method and Wiener's attack

Figure 1: Time cost of attacking RSA

As shown in Fig. 1(c), the green dots represent the time cost of Fermat's factorization while the yellow dots represent the time cost of Wiener's attack when the two methods can both successfully attack RSA. It shows that Fermat's factorization is quicker than Wiener's attack when they are both useful. However, in certain cases Wiener's attack succeeds and Fermat's factorization method fails, and vice versa.

6 Conclusions

I have implemented Fermat's factorization method and Wiener's attack for attacking RSA and tested them with different parameters. The result accords with the fact that Fermat's factorization method is efficient when p and q are close and Wiener's attack is useful when d is relatively small.

References

[1] Dan Boneh. Twenty years of attacks on the rsa cryptosystem. *NOTICES OF THE AMS*, 46:203–213, 1999.

[2] R. L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM*, 21(2):120–126, February 1978.

[3] Michael J Wiener. Cryptanalysis of short rsa secret exponents. *IEEE Transactions on Information theory*, 36(3):553–558, 1990.