Long Story Short

Cousin Wu

November 16, 2021

Section 1

Introduction

Chinese Remainder Theorem

Theorem

Suppose $n_1, \cdots, n_k$ *is pairwise coprime* $(\gcd(n_i, n_j) = 1 \ \forall i \neq j)$*, then the system of congruence equations*

$$\begin{cases} x \equiv a_1 \pmod{n_1} \\ x \equiv a_2 \pmod{n_2} \\ \vdots \\ x \equiv a_k \pmod{n_k} \end{cases}$$

has a unique solution x^* *mod* $n_1 n_2 \cdots n_k$*.*

Broadcast Attack

Suppose a plaintext is encrypted k times, where all the public exponent e are the same, and $k \geq e$. i.e. the following holds:

$$\begin{cases} m^e \equiv a_1 \pmod{n_1} \\ m^e \equiv a_2 \pmod{n_2} \\ \vdots \\ m^e \equiv a_k \pmod{n_k} \end{cases}$$

Then by Chinese Remainder Theorem, we get x^* mod $n_1 n_2 \cdots n_k$ that satisfies all the equations above. Then $x^* \equiv a_i \pmod{n_i}$ for any i, at the same time $m^e \equiv a_i \pmod{n_i}$ obviously.

But $m^e = \underbrace{m \cdot m \cdots m}_{e} \leq \underbrace{m \cdot m \cdots m}_{k} < n_1 \cdot n_2 \cdots n_k$, so $x^* = m^e$ (without mod), since the solution is unique mod $n_1 \cdots n_k$. So we get $m = \sqrt[e]{x^*}$.

Example: 韓信點兵

The setting is like so:

今有物不知其數，三三數之剩二，五五數之剩三，七七數之剩二，問物幾何？

Let's translate that to maths:

$$\begin{cases} x \equiv 2 \pmod 3 \\ x \equiv 3 \pmod 5 \\ x \equiv 2 \pmod 7 \end{cases}$$

What is the value of x?

Solution

By Chinese remainder theorem, we know that the solution is

$$x \equiv 23 \pmod{105}$$

If we further assume that $0 \leq x < 105$, then we can immediately deduce that $x = 23$ (without the mod).

Section 2

Long Story Short

Description

Here are the following operations that we can do (we can make a total of 17 operations):

1. flag: Get the flag encrypted with an AES key master_secret
2. pkey: Change the current RSA public key (No default key, and $e = 17$)
3. send: Use RSA key to encrypt arbitrary integers
4. backup: Use RSA key to encrypt the AES key master_secret

So the goal is to try to obtain the AES key in order to decrypt the flag.

Broadcast Attack?

So the first idea is maybe to launch an broadcast attack, since $e = 17$ is small. However, we have a lot of obstacles ahead of us.

1. We are not given the public modulus (even when changing the key)
2. We do not have 17 equations. (We need some for printing the encrypted flag and for some other operations)

Obstacle 1

Note that we can send -1 to encrypt. Since e is odd (Why?), and $-1 \equiv n - 1 \pmod{n}$, so encrypting -1 will yield

$$E_k(-1) = (n-1)^e \pmod{n} = n - 1$$

So we can send -1 to obtain the public modulus every time we change keys!

Obstacle 2

We need 1 operation for flag; then 1 for pkey, 1 to send -1 and 1 for backup to obtain 1 ciphertext-modulus pair. Then we can only have 5 ciphertext-modulus pairs ($3 \cdot 5 + 1 = 16$ already)

Broadcast Attack for Long Story Short

Now we have the secret key m, which is 32 bytes in length, and the RSA uses a 1024-bit public modulus (256 bytes) which will be larger than $2^{511*2} = 2^{1022}$.

This means that for each public modulus n_i, we have that $m \leq 2^{256}$ so $m^{17} < 2^{256*17} = 2^{4352} < 2^{5110} = 2^{1022*5} < n^5$.

So we really only need 5 ciphertext-modulus pairs!