Solutions Report

Modern Cryptology (CS641)

Computer Science and Engineering

Level 6

Team: **team58**
Ayush Bansal (160177)
Aman Deep Singh (15807084)
Gunjan Jalori (170283)

Date: May 16, 2020

1 Chapter 6 (RSA Encryption)

Since we can't access the server during this level, we skipped to the final puzzle of the level which is breaking the **RSA Encryption** (with small exponent) when you know some significant part of the message.
The problem statement provided to us is as follows:

- The public key (n, e) used for the **RSA Encryption**.
  `N = 84364443735725034864402554533826279174703893439763343343863260342756678609`
  `21689509377926302880924650595564757217668266944527000881648177170141755476887l`
  `28502044240300164925440505830343990622920190959934866956569753433165201951640 9`
  `51480026588738853928338105393743349699444214641968202764907970498260085751709 3`
  `e = 5`

- The information about the password and message.
  `This door has RSA encryption with exponent 5 and the password is`
  `588511908193557145472758995584417156637461398472460756192707453386570070556983`
  `787406377427753617688997008888580870506626143183054430644488980265035567576103`
  `429384907413616436962850518672602785678969919273519645573749776196447636332298`
  `966685117524322225281592140131733198556453516193938714334555505817416432 99`

1.1 Thinking about the Solution

For breaking the encryption, we will perform the **Coppersmith's Attack (Low Public exponent Attack)** on the password provided to us.

Before moving forward, let's state the famous **Coppersmith Theorem** as we are going to use it in our results.

Theorem 1.1
Let N be an integer of unknown factorization, which has a divisor $b \geq N^{\beta}$. Furthermore, let $f_b(x)$ be a univariate, monic polynomial of degree δ. Then we can find all solutions x_0 for the equation $f_b(x) = 0 \ (mod\, b)$ with

$$|x_0| \leq \frac{1}{2} N^{\frac{\beta^2}{\delta} - \epsilon}$$

in polynomial time in $\left(logN, \delta, \frac{1}{\epsilon}\right)$

And a *corollary* which is a direct implication of the above theorem.

Theorem 1.2
Let N be an integer of unknown factorization, which has a divisor $b \geq N^{\beta}$. Let $f_b(x)$ be a univariate, monic polynomial of degree δ. Furthermore, let c_N be a function that is upper-bounded by a polynomial in $logN$. Then we can find all solutions x_0 for the equation $f_b(x) = 0 \ (mod\, b)$ with

$$|x_0| \leq c_N N^{\frac{\beta^2}{\delta}}$$

in polynomial time in $(logN, \delta)$.

Now, let's get a few fundamentals out of the way, we know that:

Suppose we have the plaintext m and we wish to encrypt it using the public key (N, e), then the ciphertext c will be as follows:

$$c \equiv m^e \ (mod\ N) \tag{1}$$

Now, notice that the problem of decrypting an RSA-encrypted plaintext $c \equiv m^e \ (mod\ N)$ is the problem of finding the unique positive root $x_0 = m < N$ of the polynomial:

$$f_n(x) = x^e - c \ (mod\ N) \tag{2}$$

Under the assumption that inverting the RSA function is hard, we cannot solve this problem in general.

But if we cannot solve the problem for all $m \in \mathbb{Z}_n$, then it might be feasible for especially small values of m (as studied in class). Indeed, it is a well-known protocol failure of RSA that one can recover m in polynomial time whenever $m < N^{\frac{1}{e}}$. The reason why this attack works is simple:

Since $m^e < N$, we have

$$m^e - c = 0 \quad \text{over } \mathbb{Z} \tag{3}$$

and not just modulo N. Thus, we can simply take the e^{th} root of c in order to recover the value of m.

Now consider the following problem:

Problem 1.1
Suppose that $m = M + x$ for some known part M of the message and some unknown part $x \leq N^{\frac{1}{e}}$. Can we still recover m?

This situation occurs in the case of so-called **stereotyped messages**: Assume we already know a part M of the message which is always the same, for example M corresponds to "Good morning to everybody. Todays session-key is:". But symmetric crypto-schemes often need keys of length at most 80 bits. Hence, the above situation, where the unknown part x is smaller than the e^{th} root of the modulus N can easily occur in practice when RSA is used with small exponent e.

Let's consider a special case of Theorem 1.2, $b = N$ and $c_N = 1$, which is provided in the work of Coppersmith [1]

Theorem 1.3
Let N be an integer with unknown factorization. Furthermore, let $f_N(x)$ be a univariate, monic polynomial of degree δ. Then we can find all solutions x_0 for the equation $f_N(x) = 0 \ (mod\, N)$ with

$$|x_0| \leq N^{\frac{1}{\delta}}$$

in polynomial time in $(logN, \delta)$.

If we apply **Coppersmith's method** to the above Problem 1.1, An application of above Theorem 1.3 yields the following result.

Lemma 1.1
Let (N, e) be an RSA public key. Furthermore, let $c := (M + x_0)^e \ (mod\, N)$ be an RSA-encrypted message with known M and unknown x_0, where

$$x_0 \leq N^{\frac{1}{e}}$$

Then we can find x_0 in polynomial time in $logN$ and e.

Proof
Define

$$f_N(x) := (M + x)^e - c \tag{4}$$

which is a univariate monic polynomial of degree e with the small root x_0, $x_0 \leq N^{\frac{1}{e}}$ modulo N. An application of Theorem 1.3 proves the claim

1.2 Solution Outline

We are provided the public key in the problem - (N, e) and the encrypted password c.

At this point, we make an assumption that the password x_0 we want to recover is small i.e. $x_0 \leq N^{\frac{1}{e}}$, otherwise breaking RSA encryption is not feasible as we discussed in section 1.1.

If we assume full password x_0 as unknown and we know that it is small, then by using eq. (3), it can be found by taking the e^{th} root of c.
But if we try to find the e^{th} root of c, we are unsuccessful because it is not a perfect power of 5 for some integer.

So, we move to solving the Problem 1.1. For this we should already know some part of the password and small part will be unknown.

In the problem, we are given the string `This door has RSA encryption with exponent 5 and the password is`.
At this point, we make a second assumption that this problem is like a **stereotyped message**.

So, the initial plaintext message will be of the form:
`This door has RSA encryption with exponent 5 and the password is XXXXXXXX...`

Here, known part of the message is "`This door has RSA encryption with exponent 5 and the password is` " and the unknown part is what comes after this.

Now we established the assumptions, let's go through the steps we took to find out the value of x_0.

- Since RSA works only on integers, we will first convert the known plaintext into hex format (using the corresponding ascii values of characters).
- Convert the hex form into the corresponding integer M_0.
- Iterate from $i = 1$ to $i = 200$ (x_0 can be upto ~200 bits, by first assumption), for each iteration.
  - Left shift the integer M_0 by i to get integer M.
  - Solve the eq. (4) modulo N to get the value of small x_0.
  - If a solution exists, convert solution integer x_0 to hex format.
  - Finally, convert the hex format to ascii text and report the result.

The solution code for the problem can be found in `solve.sage`, we used the **Sage Math** libraries [2] to solve the equation by using the **Coppersmith's Algorithm** as described in Alexander May's PhD thesis [3]

The password retrieved was: `tkigrdrei`.

Hence, the complete plaintext will be as follows:
`This door has RSA encryption with exponent 5 and the password is tkigrdrei`

References

[1] Don Coppersmith. Small solutions to polynomial equations, and low exponent rsa vulnerabilities. *Journal of Cryptology*, 10:233–260, Sep 1997. pages 3

[2] Dense univariate polynomials over z/nz, implemented using ntl. http://doc.sagemath.org/html/en/reference/polynomial_rings/sage/rings/polynomial/polynomial_modn_dense_ntl.html. pages 4

[3] Alexander May. *New RSA Vulnerabilities Using Lattice Reduction Methods*. PhD thesis, University of Paderborn, 2003. pages 4