CS641: Chapter 6
Team name: team7 (chaos)

Komal Kalra (18111032) Riya James (18111054) Sristi Jaiswal (18111074)

April 14, 2019

Getting to the Cipher

The sequence of commands we used to get to the cipher:

- exit2
- *To escape the maze:* exit4 → exit3 → exit1 → exit4 → exit4 → exit2 → exit2 → exit1
- read

Deciphering the ciphertext

Since the value of e was small, we started with coppersmith's attack as it uses low exponent.

Coppersmith's theorem: Let N be an integer and $f \in \mathbb{Z}[x]$ be a monotonic polynomial of degree d over the integers. Set $X = N^{\frac{1}{d}-\epsilon}$ for $\frac{1}{d} > \epsilon > 0$. Then, given $< N, f >$, one can efficiently find all integers $x_0 < X$ satisfying $f(x_0) \equiv 0 \pmod{N}$.[source: wikipedia].
Coppersmith's theorem uses LLL reduction to find a short vector formed by the integer linear combination of a set of basis vectors.
In our case, $f(x) = (M + x)^e$, where $e = 5$ and M is the padding. We also don't know the padding. In order to find the padding, we made a file paddings.txt and wrote a list of possible paddings in it. Then we referred to coppersmith's code as given in this link: `https://github.com/mimoo/RSA-and-LLL-attacks`
We modified the program according to our values.

Encoding of the padding

We converted each character of the padding into ASCII and then converted it to binary of 8 bits. The final padding with which we got the output is
team7: This door has RSA encryption with exponent 5 and the password is -

Decoding the final output

We got this in output as the value of x

$$-1100101101011011011110110100101010101011000010110000100100101001000110001001100011101$$

Since this number was negative, we did a 2's complement of this number to find it's absolute value. Then we grouped it into 8 bits and took 80 bits from the right. We assumed the 8 bits to be ASCII values and converted it to corresponding characters, which gave us the password JHKUOOmnvs.

Program files

- *coppersmith.sage* Implements Coppersmith's attack using LLL reduction
- *paddings.txt* List of paddings used by coppersmith.sage
- *decode_password.ipynb* Computes 2's complement of the final value and decodes the password

In order to run the code, please install sage.