Project 3: Implement and Attack Plain RSA

Chaoyun Li

October 2022

The goal is to implement a plain RSA and then mount a chosen-ciphertext message recovery attack by using your implementation as an decryption oracle.

1 Chosen-ciphertext attack on plain RSA

Imagine a web application, taking as input encrypted messages by plain RSA. You can submit an arbitrary RSA ciphertext and the server will return plain message. But you can't mount a replay attack: the server keeps hashes of previous messages with an embedded timestamp to prevent submissions of same ciphertext.

Assume that you have captured other people's encrypted messages. Then you can easily recover the message by sending a maliciously crafted ciphertext to the server.

Let (n, e) be the public key. Let r be a random number modulo n such that $r \neq 1 \bmod n$. If c is the captured ciphertext, then we take

$$c' = r^e c \bmod n\,.$$

Submit c', which appears totally different from c, to the server, recovering m', which appears totally different from m. But we have

$$m = \frac{m'}{r} \bmod n\,,$$

since $c' = r^e c = (rm)^e \bmod n$ and hence $m' = rm \bmod n$.

2 Task 1

Implement plain RSA encryption and decryption. Two 1024-bit primes are provided (see attachment in Moodle). You need to choose the private key

d and then compute the public key e. You are supposed to implement the following functions:

```
def modexp(x,a,b) # compute x^{a} mod b
def rsa_enc() # given message m, return ciphertext c
def rsa_dec() # given ciphertext c, return message m
```

Note that you may need to implement a function `modinv()` to compute the modular inverse. (Hint: use Extended Euclidean algorithm[1])

3 Task 2

Implement a chosen-ciphertext attack using the `rsa_dec()` function you have implemented before. The main task is to implement the following function:

```
def attack(c,n,e)
#  given public key (n,e), ciphertext c, return message m #
```

4 Requirements

You are only allowed to used basic python libraries such as `math` and `random`. To ease the grading, please submit the source codes and write detailed comments of your codes! Please also submit a report including

- your choices of d and e
- how long does it take to do one encryption and decryption with your implementation

[1] `https://en.wikipedia.org/wiki/Extended_Euclidean_algorithm`