Lab04
Experimenting with RSA

4.1 RSA implementation (10 points)

Implement the RSA algorithm. Your program should be able to encrypt and decrypt any number from 1 to 99.

Submit your implementation file 4.1_my_rsa.py

4.2 Generating rsa-key pair and using them (10)

Generate rsa key pair using 'ssh-keygen'. Use these private and public keys to encrypt and decrypt "hello world".

Submit your code on 4.2_rsa_key.py

4.3 Breaking the RSA (20 points)

Files

1. 4.3_ciphertext.hex: ciphertext in hex string
2. 4.4_public_key.hex: public key in hex string
3. 4.5_modulo.hex: RSA modulo in hex string

Let's see if you can break the RSA. You are given a ciphertext in 4.3_ciphertext.hex that was encrypted with 1024-bit RSA using the key in 4.4_public_key.hex and the modulo in 4.5_modulo.hex. You do not know the necessary key to decrypt this ciphertext, but you do know that there is a weakness in this RSA key pair that makes it vulnerable to one of the attacks discussed in class (Please do not ask for the computation time and power consumption of our machine). The plaintext that was encrypted is a decimal integer number < the RSA modulo with a certain pattern that should be obvious when you decrypt it (The pattern is visible when the number is in its decimal representation, not hex). Your task is to find the private key and the decrypt the ciphertext to get the plaintext.