Project Description

This project expands on the uses of modular arithmetic. In this project, you will

- Understand the difference of private keys and public keys
- Create a private key and a public key for a RSA cryptosystem
- Encrypt a plaintext into a ciphertext using a RSA cryptosystem
- Decrypt a ciphertext using a RSA cryptosystem
- Implement Factorization by Squares to break into an RSA cryptosystem

Project Rubric

Each group will submit a single file. This file will be graded, out of 72 points, using the following rubric.

- 5 points for submitting the project as a completed jupyter notebook (ipynb) file.
- 5 points for legibility.
- 12 points for Questions 1-4 (3 points each)
- 12 points for Questions 5-7 (4 points each)
- 6 points for Question 7-8 (3 points each)
- 24 points for Questions 9-14 (4 points each)
- 8 points for Question 15

RSA Cryptography

Let us consider the stereotypical situation in cryptography.

Alice and Bob are having a private conversation, but they know that an eavesdropper, named Eve, could be reading their messages. In order to hide their conversation, Alice and Bob disguise their messages using a cryptographic scheme.

Cryptography is the art of securing your messages to prevent unauthorized access. When first introduced, it was used to secure messages as they were sent from one person to another. Now, cryptography provides a level of security and integrity in our modern day life of using communication devices and online services. This project will focus on the RSA cryptosystem.

What to Expect

While most of the project can be done by hand, your group is expected to code many of the project questions via Python. You will be given a notebook file for Jupyter Notebook and will be expected to fill in the required cells, along with answering questions in this document.

This project is broken into four components:

- Key Creation
- Encryption
- Decryption
- Cryptanalysis

Key Creation

Alice and Bob need to devise a system for sending secure messages back and forth. We will focus on Alice receiving messages from Bob. Since she is receiving messages, she will create a **private key**. Only Alice will know the private key, and will use this to decode any message from Bob.

If the key is constructed carefully, it becomes very difficult for Eve to understand the message. Let us go over key creation.

1. Alice will choose distinct primes p and q. If p and q are prime, there are no smaller numbers that can be used to find the private key.

   Note: Modern day computing struggles with large numbers. So in theory, choosing two large primes p and q will create a difficult problem for conventional computers to factor pq.

2. Now, Alice will create the public modulus $n = pq$. This modulus should be very large and can only be factored in to p and q. Since this is a hard number to factor, Alice feels safe posting this number to the world.

3. For future decryption reasons, Alice will compute $\phi(n) = (p-1)(q-1)$.

4. Alice will choose another integer to complicate the system, let us choose e. This integer serves as the **encryption exponent**, and Bob will use e later during encryption. Alice only needs to guarantee that $1 < e < \phi(n)$ and $gcd(e, \phi(n)) = 1$.

5. Now, Alice will keep her private key (p, q) a secret and will publish her public key (n, e).

Encryption

With the public key, Bob can now encrypt his message. His message is referred to as the **plaintext**. For the sake of the example, he wishes to send an integer m such that $1 \leqslant m < n$.

1. Bob chooses his integer m, such that $1 \leqslant m < n$.

2. To encode his message, Bob computes

$$c \equiv m^e (\bmod n).$$

3. Bob will send the encoded message c, known as the **ciphertext**, to Alice.

As we can see, Bob does not have to do any hard calculations. Bob only has to send a message according to the system that Alice has created.

Decryption

Suppose that Alice has received the ciphertext, c, from Bob. If Alice wants to know the plaintext, she needs to recover m from c.

1. Alice computes the decryption exponent

$$d \equiv e^{-1} (\bmod \phi(n)).$$

She does this by finding d such that

$$de \equiv 1 (\bmod \phi(n)).$$

2. Alice computes

$$m' \equiv c^d (\bmod n) \equiv m.$$

Note: When finding c, a calculator may have issues with c^d for very large numbers. Make sure to implement the properties of modular arithmetic into your code.

Cryptanalysis

Cryptanalysis is the science behind breaking into cryptographic schemes. This is what Eve is focused on when trying to understand Bob's message.

Eve's only goal is to factor n. The security of Bob's message is the size of n. The larger n is, the harder it is for Eve to break into the system. For this project, she will use Factorization by Squares

1. Set $x = \lceil \sqrt{n} \rceil$, where $\lceil \cdot \rceil$ is the ceiling function.
2. Compute $x^2 - n$
3. If $x^2 - n$ is a perfect square, compute $y = \sqrt{x^2 - n}$, and $n = (x + y)(x - y)$; otherwise
4. Reset x to $x + 1$ and start over at step 2.

Questions

Please refer to the jupyter notebook file and complete the questions on the assignment.