Fundamentals of Cryptography

Homework 4

Dr. Mohammad Dakhilalian

Fall 2024

Theory Part

Thoroughly review **Chapters 7** of the book *Understanding Cryptography* to confidently address the questions.

Question 1

Encrypt and decrypt using the RSA algorithm with the following system parameters:

1. $p = 7$, $q = 13$, $d = 5$, $x = 9$
2. $p = 11$, $q = 13$, $e = 7$, $x = 4$

Question 2

Compute the following exponentiation 'x^e mod m' using the square-and-multiply algorithm:

1. $x = 5$, $e = 117$, $m = 113$
2. $x = 7$, $e = 202$, $m = 123$

Show the intermediate steps for each iteration, including the binary representation of the exponent.

Question 3

Given the RSA parameters with primes $p = 43$ and $q = 19$:

1. Which of the parameters $e_1 = 45$ or $e_2 = 61$ is a valid RSA exponent? Justify your choice.
2. Compute the corresponding private key $K_{pr} = (p, q, d)$. Use the extended Euclidean algorithm for the inversion and point out every calculation step.

Question 4

An RSA encryption scheme has the set-up parameters $p = 31$ and $q = 37$. The public key is $e = 17$. Decrypt the ciphertext $y = 2$ using the CRT(Chinese Remainder Theorem).

Question 5

Answer the following question, considering the significance of mathematics in public-key cryptography:

1. What is the concept of one-way functions in public-key cryptography? Why are these functions essential for the security of cryptographic systems? Explain with an example of a one-way function used in cryptography.
2. Explain the difference between algorithms based on the discrete logarithm problem and those based on integer factorization. How do these two categories of algorithms ensure their security?

Programming Part

Question 6

The goal of this question is to implement the RSA cryptographic algorithm in Python. This will involve generating keys, testing the primality of numbers, and performing encryption and decryption.

Requirements

Your implementation should consist of the following components:

1. Key Generation

- Create a function to generate RSA keys
- Your key generation should use a secure method to ensure p and q are large primes.

2. Primality Testing

- Implement the Miller-Rabin primality test to verify the primality of numbers p and q.
- Provide a function that takes an integer and a number of iterations as input and returns whether the number is prime.

3. Encryption and Decryption

- Develop functions for encryption and decryption:
  - **Encryption**: Implement a function that encrypts a plaintext message using the public key. The message should be converted to ASCII values.
  - **Decryption**: Implement a function that decrypts the ciphertext back to the original plaintext using the private key.

4. Example Execution

- Provide an example of the RSA algorithm in action:
  - Generate a pair of keys.
  - Encrypt a sample plaintext message.
  - Decrypt the resulting ciphertext.
  - Display the original message, the ciphertext, and the decrypted message.

Deliverables

- Submit your Python code in a single `.py` file.
- Write a brief report explaining your approach and include the **screenshots** of your correct execution.