NSR/AS Lab 5 – Public Key Cryptography

Student Name: Tran Anh Thu Pham
Student ID: 103818400

TNE30009 – Network Security and Resilience
Swinburne University of Technology

103818400@student.swin.edu.au

***Abstract*— This report demonstrates the features of public key cryptography, with a focus on the RSA algorithm and the process of decrypting messages using MATLAB. Public key cryptography uses a pair of keys: a public key for encryption and a private key for decryption. The RSA algorithm ensures secure communication by the difficulty of factoring large prime numbers. The full step of the process of the RSA algorithm is explained in detail in the report. Besides, the report highlights the advantages of RSA, such as its ability to secure communications and provide digital signatures for authentication and integrity. Additionally, this report includes the MATLAB codes for decrypting the messages by determining the private key from the public key. The result section displays the decrypted messages obtained from MATLAB code execution.**

Keywords*— *Public key cryptography, RSA algorithm, encryption, decryption, prime, factor, MATLAB, digital signatures, secure communication, modular arithmetic.

I. Introduction to Public Key Cryptography

A. Public key cryptography

Public key cryptography (asymmetric cryptography) is a cryptographic system that uses a pair of public and private keys. To be more specific, the public key is shared openly and can be used by anyone to encrypt the message. The private key is secret and is used to decrypt messages encrypted with the corresponding public key. When a message is encrypted with one key, it can only be decrypted by the other key in the pair. The public key and private key are mathematically dependent. The algorithm used to generate these keys ensures that a message encrypted with one key can only be decrypted by the corresponding key. One of the most common public key algorithms is RSA which is mentioned in detail below. Public key cryptography can be used for digital signatures and key exchange.

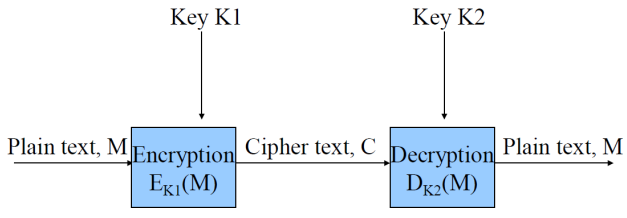

Fig. 1. Public key cryptography

Secret Key Encryption involves using the same key for both encryption and decryption processes. This key must be kept secret and shared only between the communicating parties. Some common algorithms for secret key encryption are AES (Advance Encryption Standard), DES (Data Encryption Standard), etc. On the other hand, Public Key Encryption uses a pair of different keys (K_a and K_b). The messages which are encrypted with K_a must be decrypted with K_b and vice versa. One key is made public while the other is private. One of the most commonly known public key algorithms is RSA (Rivest-Shamir-Adleman).

One of the most important applications of Public key cryptography is signing and authentication. To be more detailed, a hash of the document is calculated using a cryptographic hash function The calculated hash is encrypted using the sender's private key, creating a digital signature. The third-party can later recompute the hash from the received document. The encrypted hash which is the digital signature is decrypted using the sender's public key. The recomputed hash and the decrypted hash are compared. If there is a match, the document is protected by a hash within the signature. There are some benefits of digital signatures using Public key cryptography. The digital signatures use encryption to ensure the validity of the signature and prevent the document from changing and maintain a high level of security. Additionally, it is convenient to use digital signatures because digital signatures enable remote signing. [1]

Non-repudiation ensures a party to a digital transaction cannot deny the authenticity or occurrence of the transaction. It assures the integrity of agreements and transactions and protects other parties to contracts. In addition, Non-repudiation protects commercial interests.

In public key cryptography, the security of algorithms like RSA depends on the complexity of factoring large numbers, particularly prime factors. For instance, RSA is based on the challenge of factoring large numbers into their prime factors. E.g. Brute force attack cracks key larger than 512 bits. Therefore the recommended key length is 1024 bits for standard security requirements. For enhanced security, a key length of 2048 bits is devised.

Public key cryptography makes use of one-way functions. These functions are easy to calculate in one direction but impossible or difficult to find the inverse. For example, it is easy to calculate 52,396 x 842,412 = 44,139,019,152 but it is very hard to find the factors of 44,139,019,152.

B. RSA public key algorithm

RSA algorithm relies on the factorisation of large prime numbers ranging from 100 to 200 digits in length. While multiplying and calculating products of large numbers is computationally efficient, factoring a large number into its prime factors is significantly more challenging, especially when the number is known only as the product of two primes.

There are some advantages of the RSA algorithm. Firstly, RSA is a secure encryption method when implemented correctly with large key sizes. The reason for its security is the difficulty of factoring large composite numbers.

Secondly, RSA employs a public key for encryption and a private key for decryption. This separation ensures high security because the private key does not be shared. Thirdly, RSA supports digital signatures, authentication, integrity, and non-repudiation for data transmitted. However, there are some disadvantages of the RSA algorithm. Firstly, RSA needs large key sizes to maintain security so it can slow encryption and decryption processes. Secondly, RSA is vulnerable to quantum computer attacks which can break the RSA encryption by efficiently factoring large numbers using Shor's algorithm.[2]

The following steps are implemented in the RSA algorithm. To create the public key select two large positive primes (numbers p and q), e.g. p = 7 and q = 17. Next, compute n = p*q = 119. Then, compute x = (p-1)*(q-1) = 96. The next step is choosing an integer e which is relatively prime to x, e.g. e = 5. The public key now is [e, n] = [5, 119]. To create the private key, compute d with (d*e) mod x = 1. In this case, d = 77 and the private key is then [d, n] = [71, 119]. Assume that the data to encrypt is m = 19. To encrypt m, compute c = m^e mod n = 66. To decrypt c, compute m = c^d mod n = 19. [3]

If p*q = n then the public key is [e, n] and the private key is [d, n]. To calculate [n, d], it is essential to calculate the inverse of e in the finite field of integers mod n. Without knowing p and q, it is very difficult to calculate x and d. Therefore, p and q should be discarded when the keys are generated. Additionally, it is very challenging to determine p and q for large n. The power of the RSA algorithm is the challenge of factoring large numbers (100 to 200 digits) which are the product of two primes.

II. Breaking the RSA Algorithm

A. Premilinary

It is mandatory to download Matlab for this lab or use it online by creating an account using a student email. The next step is to download the cipher text file and decryptString.m text file from Canvas and then put it in the working directory by dragging those files from the File Explorer directly into the left-hand panel as shown in Fig. 2:

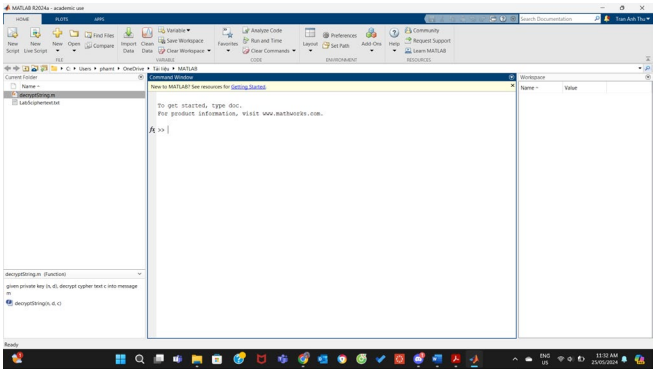

Fig. 2. MATLAB working directory

All the codes should be written in the command window and all the variables and values must be appeared in the workspace window.

B. Method

The action above decrypts a string of cipher text using the appropriate key. It is mandatory to determine the private key from the public key. In this case, the private key [n, d] associated with the public key [n, e] = [2407, 57]. It is assumed that d is less than n and d is unique. The for loop in this case is used to test different values of d.

The next step of the process is using the private key to decrypt the message. To be more specific: decyptString(n, d, c). n and d is the private key and c is a vector containing the cipher text. To obtain the full message, repeat with the public key
[n, e] = [7663, 89] and for the cipher text c in the cipher text file.

C. RSA algorithm

The full steps of the RSA algorithm have been specified in the section above; however, it is summarised below[4]:

Step 1: Compute n = p*q

Step 2: Compute x = (p-1)*(q-1)

Step 3: Choose an integer e which is relatively prime to x. Public key: [e, n]

Step 4: Compute d that (d*e) mod x = 1. Private key: [d, n]

Step 5: Assume the data to encrypt is m. To encrypt m, compute: c = m^e mod n

Step 6: To decrypt c, compute m = c^d mod n

D. MATLAB introduction

MATLAB is used for operating on matrices and vectors. There are some MATLAB commands and their purposes[4]:

- To define a matrix: *A = [1 2 3 4 5 6 7 8 9];*
- To access the vector elements: *A(i);*
- To display the value of x: *disp (x);*
- To implement the for loop:
  for count = start value : end value
  statement
  end
- To define the strings of characters: *textstring = 'a string of text'*
- To access the individual elements of the string: *textstring(i)*
- To return the prime factors of n: *factor (n)*
- If statement: *if (x==1) disp (x)*
- Return x mod y: *mod (x, y)*
- Return the length of vector x: *length (x)*
- To break the current of the for loop: *break*

There are the explanations of each line of code:

- *n = 2407;*
  e = 57;
  From here, n and e are defined. N is the product of two prime numbers which is the part of the public key and e is the component of the public key.

- *c = [2050 2296 640 479 640 2377 1274 479 640 2377 2395 194 476 2377 2395 602 2014 640 1205 2377 476 1888 2377 640 1142 1421 479 602 2014 2395 586 476 1142 749 2377 476 1142 640 2377 2395 2296 1274 2395 2377 194 586 1285 1285 2377 2014 479 640 1904 640 1142 2395 2377 602 476 540 479 2377 1205 586 1205 2395 640 479 2377 1888 479 476 2011 2377 479 640 1274 1741 586 1142 1019 2377 602 476 540 479 2377 1741 586 1274 479 602 2377];*
  c is an array of numbers that are the encrypted message. Each number in the array represents a part of the message that was encrypted using the public key [e, n].
- *g = factor(n);*
  p = g(1);
  q = g(2);
  The factor function returns the prime factors. To be more specific, n is the product of two large prime numbers p and q. Additionally, p and q are retrieved from the array g which holds the factors of n.
- *x = ((p - 1) * (q - 1));*
  *n = (p * q);*
  d = mod (e, x);
  x is the function that calculate (p-1)*(q-1). This value is used for determining the private key. In addition, n in this case is recomputed. d in this case is the modular multiplicative inverse of e mod x.
- *for i = 1:n*
  *if mod ((i*e) , x) ==1*
  d = i;
  end
  The for loop finds d that d*e mod x = 1 and it iterates through all values from 1 to n to find the correct d. When the condition mod ((i*e) , x) ==1 is met, d is set to that value of i.
- *textstring = 'Message one:';*
  The code above will display the string “Message one:” for the decrypted message output.
- *z = decryptString (n, d, c);*
  decryptString is the function that takes n, d, and c which are the encrypted messages as the inputs and then returns the decrypted message z. This function will use the private key [d, n] to decrypt the message.
- *disp (textstring);*
  disp (z);
  The two lines of code above will display the string “Message one:” followed by the decrypted message z.

The same explanation for message 2 codes.

III. Result

The Fig. 3 and Fig. 4 below are the results from running my code and the first decrypted message.

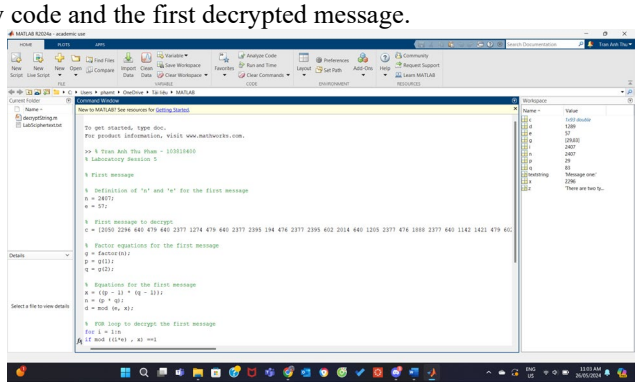

Fig. 3. First message codes

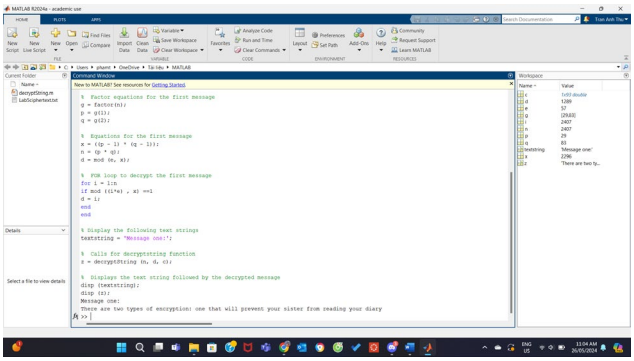

Fig. 4. First message codes (continue) and first message result

Fig. 5 and Fig. 6 below are the results of running my code and the second decrypted message.

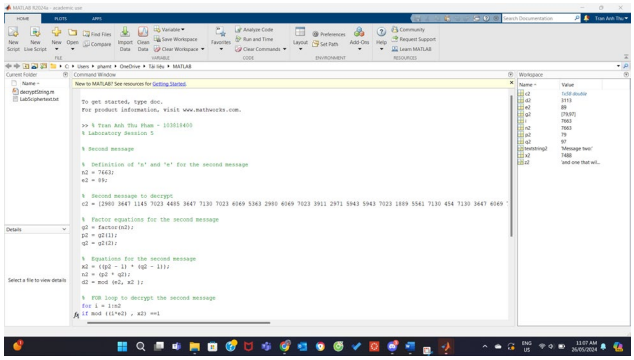

Fig. 5. Second message codes

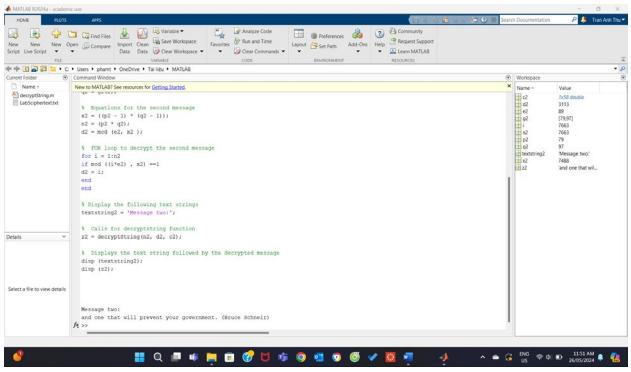

Fig. 6. Second message codes (continue) and second message result

Fig. 7 beside is the workspace after executing the running codes above.

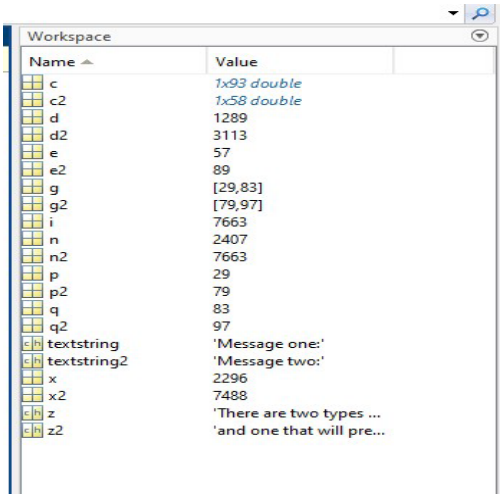

Fig. 7. Workspace after running code

IV. Conclusion

This report provided a detailed explanation and the features of public key cryptography and the RSA algorithm. The theoretical process of the RSA algorithm was mentioned with the explanation of all the steps and the difficulty of factoring large prime numbers. Additionally, this report demonstrated the process of generating RSA keys and decrypting encrypted messages using MATLAB. The RSA algorithm's advantages include securesecure message encryption, digital sugnatures for authentication, and data integrity. The MATLAB codes that include the if statement, for loop, display function, factoring function, etc illustrated how to decrypt a cipher text using a given public key and how to compute the corresponding private key.

References

[1] "The Advantages of Digital Signing: A Breakdown," www.addosign.com. https://www.addosign.com/blog/advantages-of-digital-signing#:~:text=It%20uses%20advanced%20encryption%20to (accessed May 25, 2024).

[2] "What are the advantages and disadvantages of the RSAalgorithm?," Quora. https://www.quora.com/What-are-the-advantages-and-disadvantages-of-the-RSA-algorithm (accessed May 25, 2024).

[3] "Redirecting," login.microsoftonline.com. https://swinburne.instructure.com/courses/57020/pages/lectures-week-9?module_item_id=3845001

[4] "Redirecting," login.microsoftonline.com. https://swinburne.instructure.com/courses/57020/pages/laboratory-week-10?module_item_id=3845005 (accessed May 25, 2024).