Differences between English and Chinese in the Implementation of the RSA algorithm

Duoji Jiang¹, Kwame Yankson²
¹Foreign Language School attach to Sichuan International Studies University, Chongqing, China
²School of Education, University of Michigan

Email: kyankson@gmail.com; 1013878088@qq.com

Abstract—In this paper, we explore the implementation of RSA Public Key Cryptography in both English and Chinese. We begin by examining the history of ancient and modern cryptography, public key cryptography, the key distribution problem and how the RSA algorithm solves the key distribution problem. We then show how the RSA algorithm works in practice, on both English and Chinese characters, words and sentences. We end with a comparison of differences in implementation between the two languages.

1. Introduction
Cryptography is a discipline that studies how to pass information secretly. It has been used by Royalty and by spies since ancient times. In modern times, it especially refers to mathematical research on information and its transmission. It is often regarded as a branch of mathematics and computer science. It is also closely related to information theory. Cryptography is the core of information security for related issues such as authentication and access control.

The primary purpose of cryptography is to hide the meaning of information but not to hide the existence of information, in other words, we do not hide the message but we make the message impossible to be understood without specific keys, so that a possible intruder can’t know the meaning of the information. Cryptography has been used in everyday life: chip cards including ATMs, and computer user access passwords in e-commerce as examples. The password is an important means of secrecy for the special transformation of information by both parties of communication according to the stipulated rules. In accordance with these rules, the transformed text into cipher text, known as encryption transformation; transform that back to the original text, is known as the decryption transformation.

Cryptography has gradually developed in the struggle between coding and deciphering. With the application of advanced science and technology, cryptography has become a comprehensive cutting-edge technological science. It has extensive and close relations with linguistics, mathematics, electronics, acoustics, information theory, computer science and so on.[1]

Since the 1970s, some scholars have proposed a public key system, that is based on, the mathematical principle of using one-way functions, so as to realize the separation of encryption and decryption, in which the encryption key is public and the decryption key is confidential in order to make cryptographic communication without exchanging keys possible. This new cryptosystem has aroused widespread attention and discussion in cryptography.[7]

This article focuses on one of the RSA public key cryptography algorithms based on one-way functions-RSA. The RSA public key encryption algorithm was introduced in 1977 by Ron Rivest, Adi Shamir and Leonard Adleman [2]. It was first published in the United States on July, 1987, when all three of them was working at MIT. RSA is the first letters of their surname spelled together. RSA is by far the most influential and popular
public-key encryption algorithm that can resist most of the cryptographic attacks known so far and is recommended by the ISO as a public-key data encryption standard.

Today, only a short RSA key might be broken by a powerful way. In fact, using classical algorithms, there is no reliable way to attack the RSA algorithm even today [3]. The RSA algorithm is based on a very straightforward number theory: multiplying two large primes is easy, but it is extremely difficult to factorize their product, so the product can be exposed as an encryption key. Due to the way RSA algorithm was designed, in which the encryption and decryption do not share the same key, it becomes possible to distribute public keys across the internet. Before the invention of RSA algorithm, the Key distribution problem—how to transmit encrypted message without having to transmit the secret key—had bothered people for a long time, the RSA algorithm is actually a solution to the problem at beginning, and the key distribution problem will be introduced in the next section.

As internet technology became widespread across the world, the RSA algorithm is also applied by different peoples using different languages across the world, and our topic of this paper is to find the difference of the application of RSA in two specific languages—Chinese and English.

2. The Key Distribution Problem
The key distribution problem was the most significant problem for postwar cryptographists. Before the 1970s, all of cryptography was based on symmetric functions. In a symmetric function \( f(x) = y \), if we know the value of \( x \) or \( y \), we can then compute the magnitude of the other one without too much effort. The way to get \( y \) if we have \( x \) is exactly reverse to the way to get \( x \) if we know \( y \).

In practical terms, if Alice wants to send a message to Bob using symmetric-function-based cryptography, she firstly encrypts her message with her secret-key and sends the message to Bob. Bob then decrypts the encrypted message with the same key as Alice’s, however, Alice has to make sure that Bob has the same key as hers before she sends the encrypted message to him. Otherwise the message will be meaningless words. This would not be a serious issue if Alice and Bob are friends, live in the same city and can meet discretely to exchange the key. For governments and banks that have thousands of messages that need to be securely sent to all over the world, key distribution posed a complex problem.

The problem lies in the fact that in general, if two parties have to communicate secretly with a third-party to deliver the secret key, a weak link is created in the exchange process. This is because the key can be compromised during exchange between the two parties. The risk of the secret-key being stolen during delivery cannot be ignored. This is the weak link. Otherwise it is nearly impossible for intruders to break governments and banks’ ingenious encryption algorithms even if they were symmetric-function-based. Therefore, the banks and governments before and during the 1970s had to make sure that they had the most trustworthy people to deliver keys face-to-face for them, in case of any potential intruder. The cost of doing that could be enormous.

For example, in the 1970s, banks hired trustworthy employees as full-time key carriers, these carriers traveled around the world with a locked box, delivering the keys to the clients face to face, and the clients will therefore receive encrypted documents next week. The cost of daily key-delivering increases and with the expanding of the business network, more messages had to be sent and more keys to be delivered. Finally, the cost of key distribution became a nightmare for the banks.

To some extent, militaries and governments of many countries can also spend enormous amounts of money on key distribution, simply because their messages are too significant that they need to protect the safety of these messages at any possible cost. During the Second World War, the German high command had to deliver monthly magazine with daily keys to all the operators of enigma machine-a cipher machine of Nazi German army. The secret keys of United States government were administered and delivered by COMSEC (Communications Security Establishment). In the 1970s, COMSEC delivered tons of secret keys every day.

Hellman and Diffie from Stanford University first proposed the solution to the key distribution problem. They assumed a scenario involving three people: Alice, Bob and Eve. In their scenario, Alice wants to communicate secretly with Bob, but Eve always tries to steal their keys. How can Alice interact with Bob without passing the key to Bob in person for security? Hellman and Diffie thought of a scenario where Alice loads her message in a box, then puts on her own lock and sends it to Bob. Bob puts his lock on and sends it back to Alice. Alice can remove her lock and send it back to Bob again, then Bob only needs to unlock his own padlock to get the message. Through this means of interaction, Eve has no chance at all to steal their key because the message is always encrypted while delivering. A problem that plagued people for two millennia seems to have a proposed solution.

However, in practice, key distribution is not as simple as this theory might suggest. In the hypothetical situation as mentioned, you can choose to open the lock of Alice or Bob firstly, and you can finally open the box successfully. However, for most cryptography algorithms using symmetric functions, you have to unlock in the exact reverse order as you locked them up. Take a simple example: Alice’s encryption method is to double the number. Her decryption method is to divide the ciphertext by 2. Bob’s encryption method is plus 2 on the text, his decryption method is to subtract 2. Assuming that the initial information is 2, Alice encrypted first: \(2 \times 2 = 4\). Bob then encrypted by adding 2 to get the ciphertext: \(4 + 2 = 6\). Now, in order to decrypt a symmetric-key
based ciphertext, you must follow exactly the opposite order of encryption. First reverse Bob’s encryption: 6 * 2 = 4. Then reverse Alice’s encryption: 4 / 2 = 2, in order to get the original information. If you do not follow the order, say you decrypt Alice’s encryption firstly: 6 / 2 = 3, then solve Bob’s: 3 - 2 = 1, the result you get,l, will be completely different from the original text, 2). Diffie and Hellman had to find one way that is different from symmetric key algorithm, which needs an exact reverse order of encryption to decipher a cyphertext.

To solve this problem, Hellman and Diffie used a mathematical concept: one-way functions, that are asymmetric rather than symmetric. For a one-way function, f(x) = y, it is relatively easy to calculate y when x is known, but it is extremely difficult to calculate the corresponding x if y is known. In other words, it is much more difficult to calculate the inverse. In particular, Hellman and Diffie used the modulo arithmetic idea that A (mod B) equals to the remainder when A is divided by B (See Appendix for explanation of modular arithmetic). For example 16 (mod 5)=1 because 3 times 5 plus 1 equals to 16, which means 1 is the remainder after 16 is divided by 3 times 5. When ask you the value of 16 (mod 5), it is pretty easy to get the only result -1. However, it becomes pretty complicated if you want to track back to know the original number, x with knowing x (mod 5) = 1, because x could be many numbers in this case, it only has to be positive integer times 5 plus 1, it could be 6, 11, 16... the only mean to find the original cipher is to try them all one by one. Because of the special nature of the one-way function: it simple to execute in one direction but to attack or break through it from another direction is extremely difficult, ….

It becomes the basic idea of public-key cryptography, and the detailed procedure will be introduced in the following article.

It is noteworthy that the RSA algorithm is not the first successful case of a public-key cryptography algorithm. In fact, Hellman and Diffie first proposed an algorithm of their own[5], the basic procedure is as follows:

1. Alice and Bob appoints one random number g, for example, g = 5, one prime number p, say p = 23, together and both g and p are public.
2. Alice and Bob generates their own random number, say a and b, individually and privately, for example a = 6 and b = 15.
3. Alice calculates ga(mod p), which in here equals to 56(mod 23) = 8, and send it to Bob. Bob calculates gb(mod p), in here is 151(mod 23) = 19, and send it to Alice. (Here Alice and Bob had to send each other some information before they could have the same key, which requires them to be online at the same time.)
4. Alice calculates [gb(mod p)a](mod p) with her private key a. Bob calculates [ga(mod p)b](mod p) with his own private key b. They will then successfully have the same key without tell each other the key in fact, 196(mod 23) = 815(mod 23) = 2.

This article will not spend space here explaining its proof since our main focus is the RSA algorithm. However, one thing worth noting is that there was a flaw with the algorithm: it requires both sides of communication to be online at the same time in order to share the same secret key, as mentioned in step 3, Alice and Bob had to send each other the result of ga(mod p) and gb(mod p) before they could share the same key. Diffie and Hellman couldn’t ignore this flaw.

Diffie then proposed a cutting-edge conception-asymmetric key. Simply speaking, it means the encryption key and decryption key of one algorithm don’t share the same value. The point is, Alice could then make public the encryption key so that her friends all over the world could send e-mail to her using this public key. She could keep the decryption key secret to ensure that any potential intruder wouldn’t be able to decrypt the ciphertext. Though it was a great proposal, Diffie never made his proposal come true. It was three young men from MIT-Rivest, Shamir, and Adleman who implemented Diffie’s proposal. These three young men created one algorithm that is still being widely-used even today - the RSA algorithm.

3. Prepare Your Paper Before Styling

Based on one-way function, Rivest, Shamir and Adleman implemented Diffie’s asymmetric key conception[6].

The most significant part about RSA is it used the concept of one-way functions to achieve Diffie’s asymmetric key proposal. They used some properties of prime numbers and modular arithmetic in their one-way function. In order to understand the algorithm, we have to introduce these specific properties before going into the algorithm.

3.1 Prime factorization as a one-way function

A prime number is a kind of integer that has only two factors. For example, 12 is not a prime number because it could be divided by 1, 2, 3, 4, 6 and 12. However, 17 is a prime number because it can only be divided by 1 and 17. Numbers that are not prime are called composite.

To understand the specific property of prime numbers that RSA uses, we need to understand factorization. Factorization is finding all the numbers that divide a composite number. Factorizing a big composite number could be unimaginably time consuming because the only way of factorization till today is to start to try the composite number by all the known prime numbers one by one. For example, dividing the composite number 428,207,868, which equals to 22 * 32x 112 x 197 x 499, takes a lot of time because you have to try with all the
known prime numbers one by one, and it will take a lot of time trying with smaller factors until you find the bigger factors 197 and 499.

Theoretically, when the composite number is big enough, it will take forever to find all the factors of it. However, compare to the tedious process of factorization, it is much easier to construct the number 428,207,868---just randomly choose a few big numbers on the prime number table and multiply them, you will have a big composite number that is difficult and time consuming to factorize. In other words, it is relatively easy to construct a composite number that could be difficult to factorize---an ideal one-way function.

As we can see in the RSA algorithm below, N is the product of two big prime numbers, p and q, which means N itself is considerably big too. One of the principles for RSA is, you can encrypt the message if you know N, but you can decipher it only if you know p and q. Getting N from p and q is easy, you just multiply one with another, but knowing N, finding p and q becomes much more complicated. As discussed in the previous paragraph, it is hard to divide big composite numbers, especially when their factors are relatively big like p and q here. Therefore, N could be the public key. Alice could publicize them in order to let Bob send messages to her without having to worry about a potential intruder because only she has the two numbers needed to decipher the ciphertext-p and q.

3.2 Modular arithmetic

Modular arithmetic is a process to get the remainder of a division of integers, the divisor of division is called modulus. For example, 80(mod 50) = 30 because 30 is the reminder of the division 80 ÷ 50=1 reminder 30. In modular arithmetic, calculation from one side is relatively easy for it is still essentially a division of integers. In contrast, it is more difficult to get the original number even knowing the reminder and modulus because there is an infinite amount of numbers that lead to the same reminder. For example, 80 (mod 50) = 30 because 80 ÷ 50=2 remainder 30, and so on, and in general, 30+50×N (mod 50) where N is a random integer will always equal to 30. In other words, it is an algorithm that is easy to calculate from one side but nearly impossible to calculate from the other side-another ideal one-way function.

In the RSA algorithm, M stands for the original text and C stands for the ciphertext, in the encrypting step. C is calculated by C = Me (mod n), where n and e make up the public key. To calculate Me (mod n) = C is relatively easy, it is just a deformation of division as discussed before. However, based on the characteristic of modular arithmetic discussed in the former paragraph, there could be infinite Ms even if we know e, n and C. The steps that are required to execute the RSA algorithm are outlined below:

1. **STEP 1: CREATE PRIVATE AND PUBLIC KEYS**
   First, select two (typically VERY large) and distinct primes, p and q Then compute n = pq.
   Then compute z = φ(n) = (p − 1)(q − 1)
   Then choose a number “e”, where e > 1 and gcd (e, z) = 1
   The number e is called the “public encryption exponent”
   Then find the unique number d, such that de = 1 (mod z)
   The number “d” is called the “private decryption exponent”
   Make n and e public, and keep p, q, and d secret

2. **STEP 2: ENCRYPT A MESSAGE**
   Pick a number M, where gcd (M, n) = 1 and 0 < M < n. “M” is called the message text
   Calculate C, where C = Me (mod n) using the public key (i.e. n and e).
   “C” is called the Cipher text. Send C to Bob.

3. **STEP 3: DECRIPT A MESSAGE**
   Decrypt C with the private key (i.e. d), by calculating (C)d (mod n) = M (mod n) to get the message text M.

   In order to explain how RSA works, we can work through a simple example using small prime numbers. The example is encrypting and decrypting only one letter:

   1. We firstly select two random prime numbers, here we choose two small prime numbers p=17 and q=23 in order to explain how RSA algorithm works in a simple way, but p and q are typically very big in practical use. Here n = pq=17 x 23 = 391.
   2. We then calculate \( z = \phi(n) \), which means integers smaller and relatively prime to it, and according to Euler’s phi function, equals to \((p - 1)(q - 1)=352 \). This step is important for the one-way function.
   3. We then choose another number e that is larger than 1 and relatively prime to z, which is 352. We can choose e=503 in this case.
   4. Next we find d where de=1(mod z), 503d=1(mod352), in this case we have d=7 [important for one-way function as well, for encryption and decryption].
5. Public key part: n, e as public key, keep p, q, d as private key.
6. Encryption part: Pick a message M where 0 < M < n and gcd (M, n) = 1, in this case pick 111 as our M. Keep in mind here that RSA only deals with numbers so messages to be encrypted first have to be converted into words. For our example, we encrypt by calculating \( C = M^e \mod n \), \( C = 111^{503} \mod 391 \) = 359. N and e are public keys. Alice then sends C to Bob.

7. Decryption: by calculating \( (C)^d \mod n = M \mod n \), 359 7 \( (mod 391) \) = M \( (mod 391) \), the result is M = 111.

As we can see, the asymmetric conception was perfectly achieved in the RSA algorithm, therefore Bob could send a secure message to Alice no matter whether or not Alice is online. Based on the principle of the one-way function-modular algorithm and factorization of big numbers, the security of communication was also ensured. Just as the Diffie-Hellman algorithm, with RSA, the key distribution problem was also no longer bothersome since Alice and Bob only send one-way-function-encrypted ciphertext to each other during the communication.

4. Implementing RSA – English Messages Using ASCII and Hexadecimal Code

The RSA example from the last section is for only one letter. In that example, we encrypted one letter at a time. However, in the single letter case, we still have to assign a number to the letter we want to encrypt. In the example, that number, M, is 111. Because RSA is a purely mathematical algorithm, we have to use numbers as representatives of the letters in order to encrypt the messages.

In fact, RSA can actually encrypt more than one letter at a time. RSA can encrypt an entire word by combing letters. To encrypt a sentence, we can encrypt words and punctuation. To accomplish that, we have to assign numbers to the letters and punctuation. In English, we use ASCII as numerical medium to transfer the letters and punctuation to numbers, and ASCII uses hexadecimal (see appendix for hexadecimal), range from hexadecimal 1 to ff, which is decimal 1 to 256.

For example, if we want to transmit the English message “To be or not to be?”, we firstly search for each letter or symbol’s ASCII numerical representative. \[ 54 \text{ o} 6f \text{ blank} 20 5 6 \text{ e} 65 \text{ blank} 20 5 6 \text{ f} 74 \text{ blank} 20 5 6 \text{ d} 65 5 6 3 6 5 \text{ ?} 3 f \]

Forming the words with letters, we got
\[ 5 4 6 f \text{ blank} 20 5 6 \text{ f} 6 2 5 \text{ blank} 20 5 6 \text{ f} 6 7 2 \text{ blank} 20 5 6 \text{ f} 6 6 f 5 6 7 \text{ blank} 20 5 4 6 f \text{ blank} 20 5 6 \text{ f} 6 2 5 5 6 3 f \]

Then we combine the words and punctuation together as a sentence, which is 5 4 6 F 2 0 6 2 6 5 2 0 6 F 7 2 0 6 e 6 f 7 4 2 0 6 2 6 5 3 f , use this number as M and repeat the RSA procedure, we can successfully encrypt the sentence.

5. Implementing RSA – Chinese Messages Using Hexadecimal

In order to transmit messages with languages other than Latin-letters-based languages, we use Unicode as their numerical representatives. In order to include numerous languages around the world, the Unicode ranges from 1 to 65535 in decimal format.

In Chinese, there is no letters forming the words. The characters, as the basic units of Chinese written language, directly form sentences with punctuation.

For example, if we want to transmit the Chinese message “生存还是毁灭?” (means “To be or not to be?”), the first step is to find each character and symbol’s numerical representative in Unicode.

生:751f存:5b58还:8fd8是:662f毁:6bc1灭:706d?003f

Combining together they form Hexadecimal 751f5b58fd8662f6bc1706d003f, and then we just replace M with this number and repeat the process of RSA, we can implement RSA with this Chinese sentence.

6. Discussion And Conclusion

This paper aimed to provide a comparison of the application of the RSA algorithm in Chinese and English. Indeed, RSA algorithm is mathematics-based. Messages in English and in Chinese languages have to be translated into numbers firstly and only then after can we apply the RSA algorithm. However, because of the different characteristics of the two languages themselves, the application of RSA in the two languages are different. There are mainly two differences.

The first difference has to do with the process of constructing sentences. In English, the translation happens letter by letter, the letters make up words and the words then build up the sentences while in Chinese, the translation happens character by character and the characters themselves directly compose sentences, so the application of RSA in Chinese is slightly more troublesome because it takes one more step than in English to form letters to words.

The second difference involves the length of representatives of basic units of written Chinese and English. This second difference derives from the difference of phonography and ideography between the two languages. Latin letters which English and many other western European languages use are based on phonography. In phonography, the alphabets were designed to represent sound therefore the number of basic units of
phonography is comparatively limited because of the limitation of variety of pronunciation of human language. There are only 107 letters, 52 diacritics and 4 prosodic marks in the International Phonetic Alphabet 2005 edition and only 26 letters in English.

Chinese characters on the other hand are represented as ideographs. The characters were designed to represent ideas and have little connection with the pronunciation. Compared to the simplicity of pronunciation, there are too many complicated ideas people want to express and the complicity of human thinking generated the impressive amount of Chinese characters. There are 106,230 different characters in The Dictionary of Chinese Variant Form compiled by the Taiwan(ROC) Ministry of Education in 2004. So apparently the amount of numbers to represent letters and characters are different for English and for Chinese. The ASCII has only 128 different symbols in total, two-digits long in hexadecimal. However, the part of Unicode to represent Chinese characters are four-digits long in hexadecimal which will make the calculation more complicate.

However, the RSA algorithm is a pure-mathematical algorithm. The differences in implementation in English and Chinese only lie in the differences of languages themselves. To computers, the processes of application of RSA in English and Chinese are virtually the same. At the basic level, it is logic and mathematics. Maybe this last assertion can provide us with some clues to the beauty of mathematics. It is also one of the reasons why so many talented people spend their life time study mathematics: its precision and logic go beyond all divergence and differences in human society. It is blind to all different intentions, ideas of good and evil, and racial differences among people who study and apply mathematics.

7. Appendix
Mathematics in RSA encryption and decryption
1. Prime numbers
Prime number is one kind of integer that only has two factors. A factor of a number divides that number exactly. For example 17 is a prime number because only 1 and 17 could exact divide it. However 15 ÷ 5=3, so 3 and 5 are factors of 15, and this also proves that 15 is not a prime number.. Integers that are not prime numbers are called composite numbers.

2. GCD
GCD simply means Greatest Common Divisor. Divisor and factor are synonyms. They have the same meaning. If A divides B exactly, A is B’s divisor or factor. For example, 20 has divisors 1, 2, 4, 5, 10, 20 and 24 has divisors 1, 3, 4, 6, 8, 12, 24. 1 and 4 are common divisors of both 24 and 20, therefore 4 is their greatest common divisor. Some countries say greatest common factor instead of greatest common divisor. When two integers share 1 as their GCD, those two numbers are relatively prime to each other.

3. Factorizing
Factorizing means to find one number’s factors. Unfortunately, although there appears to be certain characters with small factors, we still have to try all the prime numbers one by one to find all of a number’s factors. That is the reason we could view factoring multiple two big prime numbers as a one-way function. Multiplying is quite easy but finding the factors of their product is much more difficult.

4. Modular arithmetic
Modular arithmetic is actually in our daily life., Time is the most apparent example. For example, it is 1:30pm now, it should be 2:20pm after 50 minutes instead of 1:30. Because apparently minutes goes from 0 to 59 only, when it comes to 60 it will jump into another hour, so when 30 minutes have passed from 1:30 it has already jump into 2:00. Modular arithmetic means you divide X by Y, you get the remainder after it is divided. For example, in this case,30 plus 50 is 80, and you divide 80 with 60, you get the result 1 and the remainder 20,80(mod 60)=20 because 80=1 x 60+20. Modular arithmetic is a cycle, for example, 20 here can appear infinite times,140(mod60) also equals to 20 because 140=2 x 60 +20, in other words, if we know the value of a and b and the equation x=a(mod b) and we want to find x, it becomes quite difficult, because x could equal to any only if a + bc only if c is a integer, there could be infinite amount of values for x, that makes modular arithmetic a one-way function.

5. Euler’s phi function
\( \Phi(n) \) is the number of all the integers smaller than a positive integer N and relatively prime to it (means they only share 1 as their only common divisor). For example, \( \Phi(8)=4 \), because 1, 3, 5, 7 are smaller than 8 and relatively prime to it.

Euler’s phi function states that when a, n are positive integers and GCD (a, n) = 1, a \( \Phi(n) = 1 \mod n \)

Let n and a be positive, relatively prime integers. Then \( a \Phi(n) = 1 \mod n \).

6. Fermat
It is actually a special case of Euler’s phi function. When n is prime, \( \Phi(n)=n-1 \) because prime numbers are relatively prime to all integers besides 1.

So when GCD \( (n, a) = 1 \), a \( a(n-1) = 1 \mod n \).
7. Chinese Remainder Theorem
    Let \( m, n \) be integers satisfying \( \gcd (m, n) = 1 \). Then the simultaneous congruence \( x = b \mod m \) and \( x = c \mod n \) have exactly one solution for \( 0 < x < mn \).

8. Number Systems.
    For any \( X \)-digits number system, when the digit comes to \( x \), plus 1 to the higher digit place and the number on the original number position goes back to 0. For example, in decimal number system, when 9 plus 1, adding 1 to the tens digit and the units digit goes back to 1, which is 10.
    This becomes abstract to understand when it comes to number systems other than decimal. For example, in the hexadecimal format, we use a to f to represent 10 to 15, when 9 plus 6, it becomes f (Decimal 15) instead of 15 because there could be 16 digits in the units digit! FF in hexadecimal format equals 15 x 15 + 15 = 255 in decimal format.

8. Acknowledgment
    We would like to thank Illume.

9. References
[1] Luciano & Prichett (1987) Cryptology - from Caesar ciphers to public key cryptography
[2] Singh, S. (2000). The code book: the science of secrecy from ancient Egypt to quantum cryptography. London: Anchor
[3] Gardiner, M (1977). A new cipher that would take millions of years to break. Scientific American, 237(8), 120 - 124.
[4] Unicode Consortium. C0 Controls and Basic Latin.(2018)http://unicode.org/charts/
[5] Diffie & Hellman (1976) New directions in cryptography
[6] Rivest et al. (1978) A method for obtaining digital signatures and public key cryptosystems
[7] Hellman (1979) The mathematics of public key cryptography