# A Brief History of Cryptography

Shmuel Lotsvin<br>
640:436 – History of Math<br>
May 1$^{st}$, 2017

## 1.1 Introduction

*Cryptology* is the "study of secret writing," according to Knox University professor John F. Dooley[1]. While it is often used interchangeably with the terms *cryptography* and *cryptanalysis*, cryptology actually encompasses both of these fields. *Cryptography* is the creation of cryptologic systems and encoding of messages using these systems, while *cryptanalysis* is the process of decoding such messages, often without being told the key. Despite this, I will be using all three terms without differentiation, as they all refer to the same activity of encrypting and decrypting messages. Crpytology should not be confused with *steganography*, however, which is the act of hiding the message altogether. Cryptology does not attempt to hide the message; rather, only the meaning of the message is hidden through a series of transformations on the text itself.

Cryptography dates back to about 4000 years ago, when the Egyptians were found to be using obscure hieroglyphs in order to seemingly abstract the meaning of messages in the tombs of the dead[2]. Around 500 B.C., the Spartans were found to have used a *scytale* in order to encrypt their messages: simply wrap leather around a pole, write the message across the leather, then unwrap it, and the message will transform into its encrypted form[3]. From then on, cryptography was found across various regions in the world, and in various forms. Before I discuss several historically significant cryptographic algorithms, I must first define a few terms.

## 1.2 Definitions

*Encryption* is the process of tranforming plaintext into ciphertext through the use of *ciphers*, or algorithms; *decryption* is the process of transforming ciphertext back into its original plaintext. *Plaintext* is the original message that is being encoded using the cipher, while the *ciphertext* is the encoded message. There are two different types of cryptosystems, namely *symmetric cryptosystems* and *asymmetric cryptosystems*, both using *keys* in order encode or decode messages. A *key* can be thought of as number, function, or permutation that is used in order to encrypt or decrypt a message. *Symmetric cryptosystems* are cryptographic algorithms that require a secret key in order for one party to encode and another party to decode a message, and the bulk of this paper will discuss these algorithms[4]. On the other hand, *asymmetric cryptosystems* are algorithms which use two or more keys, generally one public and one private, and I will specifically discuss RSA, a well-known asymmetric cryptosystem.

## 2 Symmetric Cryptosystems

All historical cryptosystems that were used prior to the 1960s were symmetric cryptosystems, requiring a secret key to be shared between the encoder and the decoder[4]. I will give a brief description of several different symmetric cryptosystems that were used throughout history, before proceeding to asymmetric cryptosystems.

## 2.1 Caesar Shift

One of the most well-known symmetric algorithms is the *Caesar shift*, also know as the *Caesar cipher*, or the *shift cipher*. The Caesar shift was named after Julius Caesar, who used this algorithm to encode his messages to his military generals and congressmen around the 1st century B.C.[2] The shift cipher is fairly easy to understand: given an alphabet and a plaintext message, person A chooses a number of letters and a direction to shift the entire message, which then transforms it into a completely new, encoded ciphertext. To decode the ciphertext, person B uses the key provided by person A (in this case, the number of letters shifted and the direction of the shift) in order to decode the ciphertext back into the original message.

Assuming we use the current English alphabet, we can transform each letter in a plaintext or ciphertext using two equations:

$$E(x) \equiv (x + n) \bmod 26, \qquad D(x) \equiv (x - n) \bmod 26,$$

where $E(x)$ is the encryption function, $D(x)$ is the decryption function, $x$ is the position or value assigned to a letter in the alphabet (i.e., $a = 0$, $b = 1$...), and $n$ is the key (the number of letters shifted). To denote whether the shift is happening to the right or to the left, we let $n$ be positive or negative, respectively. Additionally, we must have *mod 26* to loop around our alphabet. For example, assume our key is $n = 3$. Applying this key to the message "xyz" will not work, because we cannot shift these letters any further to the right of the alphabet. Thus, we use *mod 26* in order to map these letters back into a letter of our alphabet; otherwise, we would never be able to encode "xyz" and we would never see "abc" in our encoded message, making the ciphertext less secure.

As a simple example of how the cipher works, assume the alphabet is the English alphabet, $n = 3$, and the plaintext message is "abstxy". After assigning each letter a value based on its position, we apply the equation above on each letter until the entire plaintext is encoded. The resulting ciphertext is "devwab", and this can be easily decoded through the use of the decryption equation above.

Unfortunately, in terms of security, the shift cipher alone is not very reliable, as the number of unique shifts in the alphabet, specifically in the English alphabet, is 26. Thus, other symmetric cryptosystems have been used instead of, or on top of, the shift cipher.

### 2.2 Substitution Cipher

The substitution cipher was another transposition cryptosystem, where the letters in the plaintext were mapped to different letters in the alphabet based on a private key. Consider a permutation of the English alphabet; there are *26! - 1* different permutations possible, not including the standard alphabet. Using this permutation, person A transforms the plaintext into the ciphertext by mapping each letter from the standard alphabet to the new letter in its position in the permuted alphabet. In this way, there is no clear pattern as to how each letter is encoded. To decrypt the message, person B uses the permutation given by person A (the private key) in order to map the encoded letters back to their original letters.

To better understand this, consider the English alphabet and its permutation below:

A B C D E F G H I J K L M N O P Q R S T U V Q R S T U V W X Y Z
**D C B A** E F G H I J K L M N O P Q R S T U V Q R S T U V W X Y Z

Essentially, the English alphabet has been permuted so that the first four letters have been reversed. This new permutation will be our key, and will allow us to encode our message. Suppose our message is "abefxy" and we use our permutation to encode the message. The resulting ciphertext is "dcefxy", which can easily be decrypted by mapping the encoded message back to the original alphabet using the same permutation.

However, the substitution cipher can be easily cracked through a simple comparison of letter frequencies. The frequency of letters appearing in an alphabet can be calculated by summing all of the occurrences of each letter in a given text, such as *Moby Dick* or *A Tale of Two*

*Cities*, and then dividing this sum by the total number of letters within the text. Because the books we use to calculate this frequency have hundreds of thousands of letters, the sample size is large enough to represent the distribution of letters in the alphabet across all English texts. Thus, we may find, for example, that ‘E’ has a 13% frequency of appearing in the use of the English language. Using this, we can compare the frequencies of each English letter to the frequencies of the encoding of the message. If an encoded letter in the ciphertext, say ‘X’, has the same or a similar frequency as another letter in the English alphabet, say ‘A’, then we can conclude that ‘X’ is most likely the encoding for ‘A’. By doing this for all letters in the encoding, we can even derive the permutation that was used to encode the plaintext, and thus decode the encrypted message without ever actually receiving the key. This method was formally mentioned by Al-Kindi, an Iraqi mathematician, in the 9th century A.D., and could be applied to decrypting the shift cipher as well[3].

During the 1800s, cipher designers attempted to create new algorithms to encode messages that would not be broken by a simple letter frequency comparison[4]. Previously, monoalphabetic ciphers, where only one permutation of the alphabet was used, were prominent. However, several polyalphabetic systems were created in order to fix the deficiencies of monoalphabetic ciphers, which included the Vigenère cipher and the Permutation cipher. The Vigenère cipher, invented by Giovan Batista Belaso in 1533, attempted to do this[4]. Rather than using one permutation of the alphabet to encode the entire message, the Vigenère cipher uses several. First, person A chooses a word, such as “PHONE”, and repeats that letter over and over so that each letter in the plaintext message is assigned some letter in the word phone:

H E L L O M Y N A M E I S S A M
P H O N E P H O N E P H O N E P

For each letter in “PHONE”, we pick a new permutation of the English alphabet; in this case, we will have five different permutations. We apply each of these permutations only to the letters in the plaintext that are paired with that letter, that is, we apply the permutation associated with ‘P’ to ‘H’, ‘M’, ‘E’, and ‘M’ again. Note that it is possible for two letters in the plaintext to be encoded into different letters, as they may be encoded using different permutations. This new level of difficulty in encoding also provides a level of security against unauthorized decoders. Unfortunately, the Vigenère cipher only delayed, but did not stop, the use of letter frequencies to decode messages, and governments continued to use this cipher for more than 200 years (even calling it the “undecipherable cipher”) before letter frequencies were calculated and utilized[1]. Further improvements and layers could be added, such as choosing a new permutation for each letter, and this was the basis of how the famous Enigma cipher operated in World War II[4]. Yet, once again, this type of cipher was decoded using cryptanalysis.

### 2.3 Permutation Cipher

The permutation cipher is another symmetric cryptosystem with various holes in security, but it is one that has been used for centuries and is still in use today. Suppose person A wants to encode a message “it was the best of times, it was the worst of times” to send to person B. To encrypt the message using the permutation cipher, the person first removes all of the punctuation and spacing between the words in the message, and then splits the message into groups of $n$ letters. Note that this choice of $n$ affects the number of possible permutations that can be chosen, and as such, larger choices of $n$, up to a point, provide more permutations and more security. If the chosen $n$ does not evenly divide the total number of letters in the message, then we can simply pad the end of string and use that as part of the encoding as well. Once the message is split into groups, person A decided on a permutation with which to permute each group, and applies it. Then, the groups of letters are recombined into one long ciphertext, which is sent to person B, along with the key (the permutation of each group of letters). Person B knows how many letters are in each group, so splitting the ciphertext and mapping the encoded message back into plaintext is simple. The steps of the permutation cipher are shown below:

1. it was the best of times, it was the worst of times (original text)
2. itwas thebe stoft imesi twast hewor stoft imesz (split into groups of 5, padded with z)
3. Let $\begin{pmatrix}12345\\52341\end{pmatrix}$ be the permutation used on each group (the first and last letter swap places)
4. stwai ehebt ttofs imesi twast rewoh ttofs zmesi (permuted groups)
5. stwaiehebtttofsimesitwastrewohttofszmesi (final ciphertext to be sent)

### 2.4 Conclusion

Despite the disadvantages, symmetric cryptosystems are still used in a variety of fields. Both the substitution cipher and the permutation ciphers are still in use today. The Advanced Encryption Standard (AES), also known as Rijndael, is a symmetric cryptosystem that works by using substitution ciphers and permutation ciphers on top of one other (also known as a substitution-permutation network). Since its introduction in the 1990s, it has been widely adopted and used for security purposes; even US federal organization such as the National Security Agency (NSA) use this system[4]. By alternating substitution ciphers with permutation ciphers, the encoded messages produced by AES are practically impossible to decode without knowing the specific sequence and the key to each individual cipher. Furthermore, elliptic curve cryptography (ECC), which based encodings on the curve of an elliptic function, is another new innovation in symmetric cryptosystems. While not as widely accepted as AES, ECC is another viable alternative to an asymmetric security system that requires more than one key.

## 3 Asymmetric Cryptosystems

Asymmetric cryptography, also known as public-key cryptography, involves the use of two (or more) keys in order to encrypt and decrypt a message. Generally, person A will encrypt their message with a public (available for all to see) key, and person B will need person A's corresponding private key in order to decrypt the message. Such cryptography systems did not appear until the late 1960s and did not gain traction until the 1970s. In 1969, James Ellis created public-key cryptography, and in 1973, Clifford Cocks implemented a cryptosystem that was basically the RSA system we know today, a full four years before its official inception[4]. In 1976, Whitfield Diffie and Martin Hellman published their work on public-key cryptography, known as the Diffie-Hellman key exchange (or Diffie-Hellman-Merkle, named after Ralph Merkle who influenced their work) and revolutionized the field of cryptography[2]. A year later, in 1977, Ron Rivest, Adi Shamir, and Leonard Adleman created what was known as the RSA algorithm, which used the premise of factoring large numbers in order to encode messages[2].

## 3.1 RSA Algorithm

The basis behind the RSA algorithm is the difficulty of factoring large prime numbers. Through the use of one-way functions, that is, functions that can only be reliably computed in one direction, RSA is able to encode and decode messages in a secure manner. RSA is a fairly simple algorithm, and the difficulty comes from the fact that factoring the product of two large primes is near impossible if the numbers are large enough. The RSA algorithm involving two people, A and B, is given below:

1. Person A secretly chooses two large prime numbers $p,q$
2. Person A finds the product of $p$ and $q$, calling this $N$
3. Person A finds the product of $(p-1)(q-1)$, calling this $N'$
4. Person A finds a prime number $e$ that is relatively prime to $N'$, that is, $gcd(N', e) = 1$
5. Person A notes down and makes available their public key, $(N, e)$
6. Person A applies the extended Euclidean algorithm on $N'$ and $e$ to obtain $d$, such that $e \cdot d \equiv 1 \ (mod\ N')$
7. Person A notes down their private key, $(d, p, q)$
8. Person B takes Person A's public key and represents the message $M$ that B wants to send as a number $m$, such that $0 \leq m < N$
9. Person B computes $c$, the ciphertext in the form of a number, such that $c = m^e \ (mod\ N)$
10. Person A receives $c$, and can decrypt this message by calculating $m = c^d \ (mod\ N)$

The reason why decryption is possible is due to Lagrange's Theorem, which allows us to write: $x^{(p-1)(q-1)} \equiv 1 \ (mod\ N)$, implying that for some integer $s$, we have $ed - s(p-1)(q-1) = 1$, which leads us to $c^d = (m^e)^d$, which is easily solvable[4].

While RSA is a fairly secure way of encrypting a message, the definition of "large enough" numbers is somewhat vague. Currently, 500-bit numbers (that is, $2^{500}$)are the largest numbers that have been factored[4]. Thus, 1024-bit, 2048-bit, and 4096-bit prime numbers should be used in order to avoid a successful decryption by an unwanted party. Unfortunately, it is hard to predict how processing power may grow in the future, so future computer systems may be able to factor these large primes without trouble. Thus, RSA is not future-proof. However, it is still a good model for other public-key algorithms, such as those used in Transport Layer Security (TLS) and Secure Socket Layer (SSL) protocols, which provide security for communication over networks like the Internet.

## 4 Conclusion

Cryptography, and by extension cryptanalysis and cryptology, has become a large field of study that is constantly changing and adapting to new technology. With the creation of faster computers and increased research in mathematics and computer science, cryptography will continue to evolve. According to Simon Singh, cryptography expert and writer, "if quantum computers were to become a reality, then RSA and all other modern ciphers would be useless, and quantum cryptography would become a necessity in order to overcome the privacy gap"[2]. While this may be somewhat infeasible today, it may be the case that in the near future, neither symmetric nor asymmetric cryptography is useful, and only quantum cryptography will be viable. However, for now, public-key and private-key algorithms are a mainstay in our current computer security protocols.

# Works Cited

1. Dooley, John F. *A Brief History of Cryptology and Cryptographic Algorithms*. Cham: Springer, 2013. Print.

2. Singh, Simon. *The Code Book*. New York, NY: Delacorte, 2003. Print

3. D'Agapeyeff, A. *Codes and Ciphers*. S.l.: Hesperides, 2008. Print.

4. Smart, Nigel P. *Cryptography: An Introduction*. London: McGraw-Hill, 2003. Print.