Assignment 1 : Encryption and decryption

Nathan Lesourd

15 Septembre 2023

Abstract

In this study we are going to study the strengths and weaknesses of two algorithms, Vigenere encryption and row transposition encryption, as well as their combination. In order to measure their strength, we will look at the success time of a brute-force attack, with knowledge of the decryption algorithms and key size. To complete this study, we will look at the effect of key length on decryption success time. Thanks to this metric, we have been able to see that the classic transposition and substitution algorithms used alone are not convincing and are vulnerable to brute-force attacks. The combination greatly strengthens the security of the encryption, but with the previous metrics it was not possible to identify the best combination. A frequency analysis was therefore carried out. In the end, it was the combination of transposition row encryption and vigenere encryption that offered the best security.

1. Introduction

Encryption algorithms are all around us these days. All the communications we can make with our phone are encrypted to guarantee the confidentiality and authenticity of the communication. In order to understand how message encryption works, we are going to study two encryption algorithms. These are much more basic methods than we find today, but they will help us to understand the mechanism. We will be looking at vigenere encryption and row transposition cipher. These two methods are symmetrical because they require a single private key. The aim is to identify the strengths and weaknesses of these two methods.

2. Design and Implementation

2.1 Part I

The implementation of the vigenere algorithm is available in the file `A1.py`. First, we change the plaintext to lower case. Then we concatenate the key as many times as necessary to make it the size of the plaintext. Next, we iterate over each character in the plaintext and the key, then add their alphabetical positions to obtain the new position in the alphabet for the encrypted character. To add these positions we use the `ord()` function, which gives the position in the ASCII table of the character in question. Check that the position in the ASCII table does not exceed 122 (ASCII table value of z). If it does, subtract 26 from the position to obtain a letter. Finally, to go from the position in the ASCII table to the encrypted character, we use the `chr()` function. At the end of the loop we obtain the ciphertext.

The advantage of Vigenere encryption is that it is easy to implement and very light. Nevertheless, the languages we use follow distributions that determine the frequency of appearance of a character. The problem is that despite the encryption of the plaintext, the distribution of the ciphertext is similar to the distribution of the original language of the plaintext. By performing a frequency analysis, we can easily find the size of the key used and then carry out a brute-force attack.

With the key's length we could perfom a brute-force attack easily with the different combinations available for this size. On average, half of all possible keys must be tried to achieve success. Therefore, we are going to compute all the possibilities and see the amount of time to do decryption process with half of all possible keys. Thus, we get the average time to achieve success using brute-force attack. This time will be a good metric to know the strength of this encryption.

We could improve this method in increasing key's length to be automatically at the size of the plaintext. Thanks to this bigger key, the number of combinations will be higher, thus brute-force attack will be longer. We could also add other layer of transformation on the plaintext in order to increase security.

2.2 Part II

The deciphering function is based on the same philosophy as the encryption function, but instead of adding the alphabetical positions, we subtract them. This time we have to be careful that the subtraction is not below 97 (the position in the ASCII table of a). If this is not the case, we add 26 to obtain a letter of the alphabet.

The order of encryption does matter when decrypting ciphertext. You have to decrypt in the reverse order of the encryption phase. For example: Encryption

1. vigenere_cipher
2. row_transposition_cipher Decryption
3. row_transposition_decipher
4. vigenere_decipher

Adding layers of encryption makes the combination much stronger against potential attacks. In fact, adding layers increases the snowball effect. In other words, a small difference in input creates a larger difference in output. As a result, understanding the encryption algorithm is far more complex and harder to attack. Even with knowledge of the combination of encryption algorithms (which is difficult to find), a brute-force attack will take much longer because there will be many more possibilities to test.

2.3 Part III

Reversing the combination produces a new encrypted result. We will see later in this report which combination provides the best security.

The row transposition cipher is a technique that allows the initial characters of the plaintext to be mixed. The disadvantage is that we know all the characters that make up the plaintext. Using row transposition cipher alone is not very effective. It is best used in combination with Vigenere encryption. The disadvantage of Vigenere encryption is its vulnerability to frequency analysis. The row transposition cypher adds a layer of ambiguity that makes frequency analysis more complex.

2.4 Part VI

From the success time of a brute-force attack, it is impossible to identify one combination that is better than another. I therefore carried out a frequency analysis on the outputs of the two combinations to compare the distributions. It turns out that the output distribution of the Vigenere then Row transposition combination is much better because it follows a distribution that approaches a uniform distribution. As a result, it is much more difficult to make a link between input and output and deduce the key sizes used. In the end, Vigenere encryption followed by Row transposition encryption is the best solution.

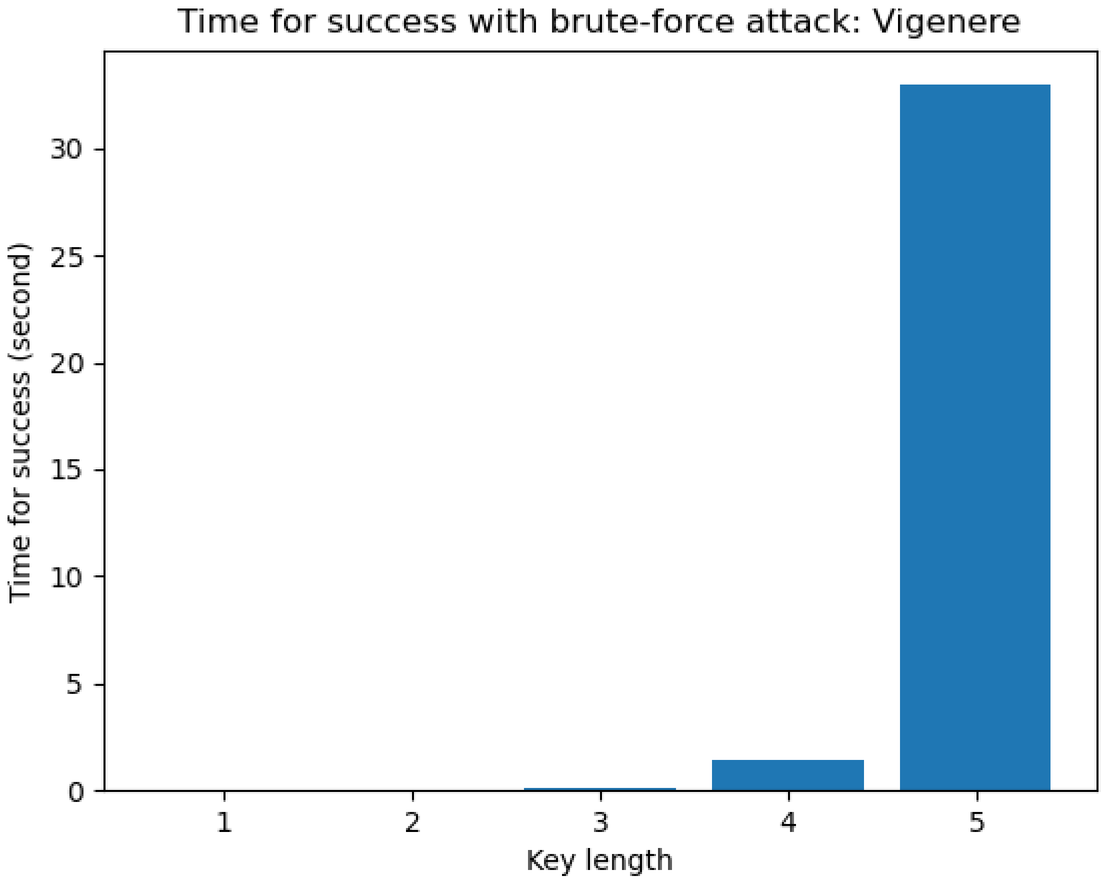
To improve the encryption process we can increase the size of the keys. We can imagine that the size of the keys can be automatically matched to the size of the plaintext. We can also multiply the number of layers by alternating transpositions and substitutions. The security of private keys is also very important. The way they are stored and generated determines the overall security of the encryption system.

3. Test Results

Vigenere cipher only

Setup: key_size_vigenere = 3

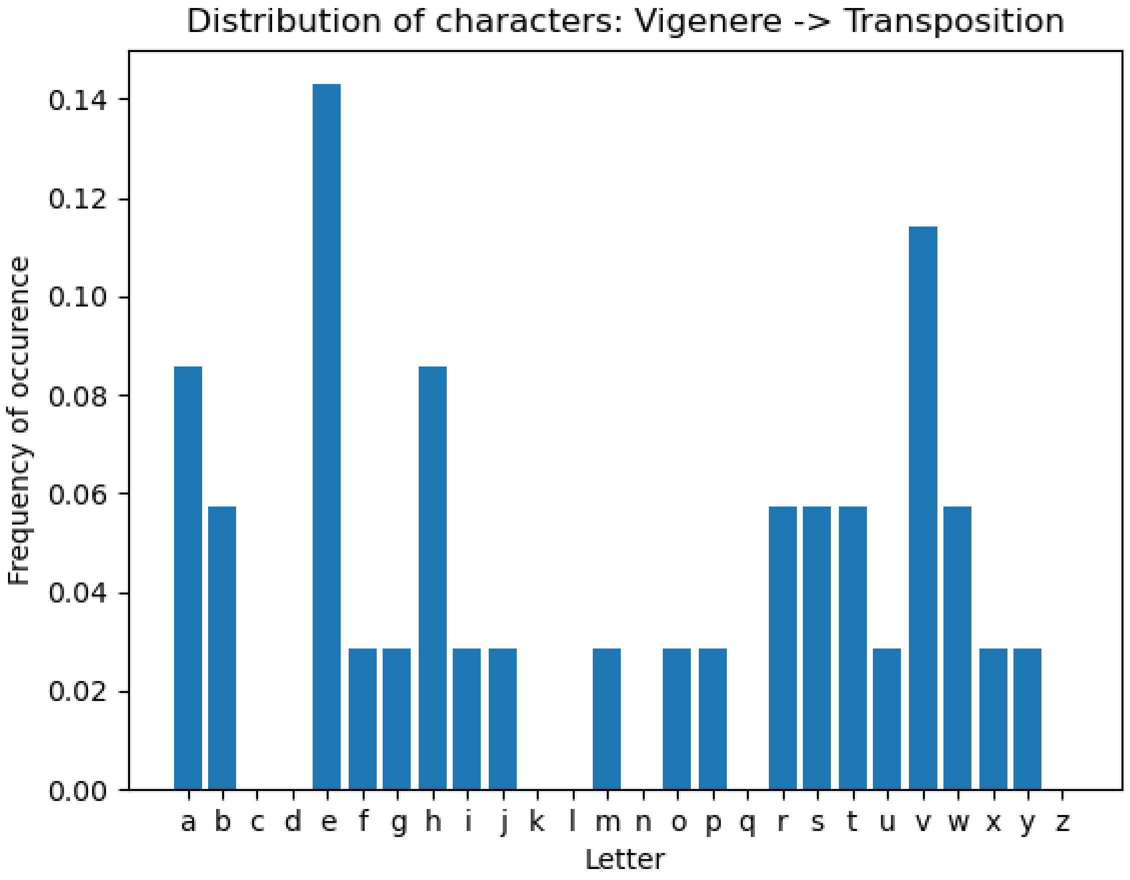
Number of combinations available: 15600 Half of all possibilities to get the average time of success: 7800 Delay to decipher vigenere encryption with brute-force attack: 0.072662 seconde(s).



Vigenere cipher then row Transposition cipher

Setup: key_size_vigenere = 3, key_size_row_transposition = 5

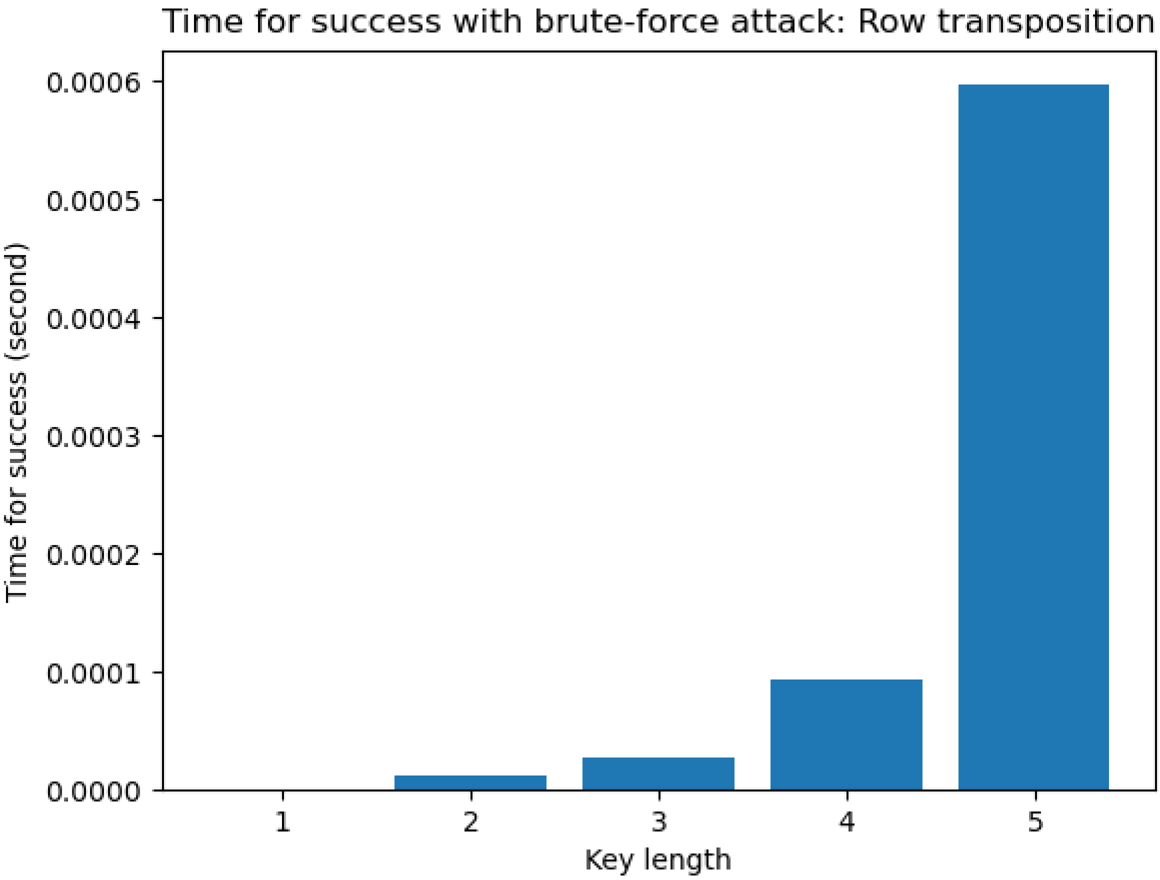
Number of combinations available: 1872000 Half of all possibilities to get the average time of success: 936000 Delay to decipher combination of encryptions with brute-force attack: 13.362344 seconde(s).



Row transposition cipher only

Setup: key_size_row_transposition = 5

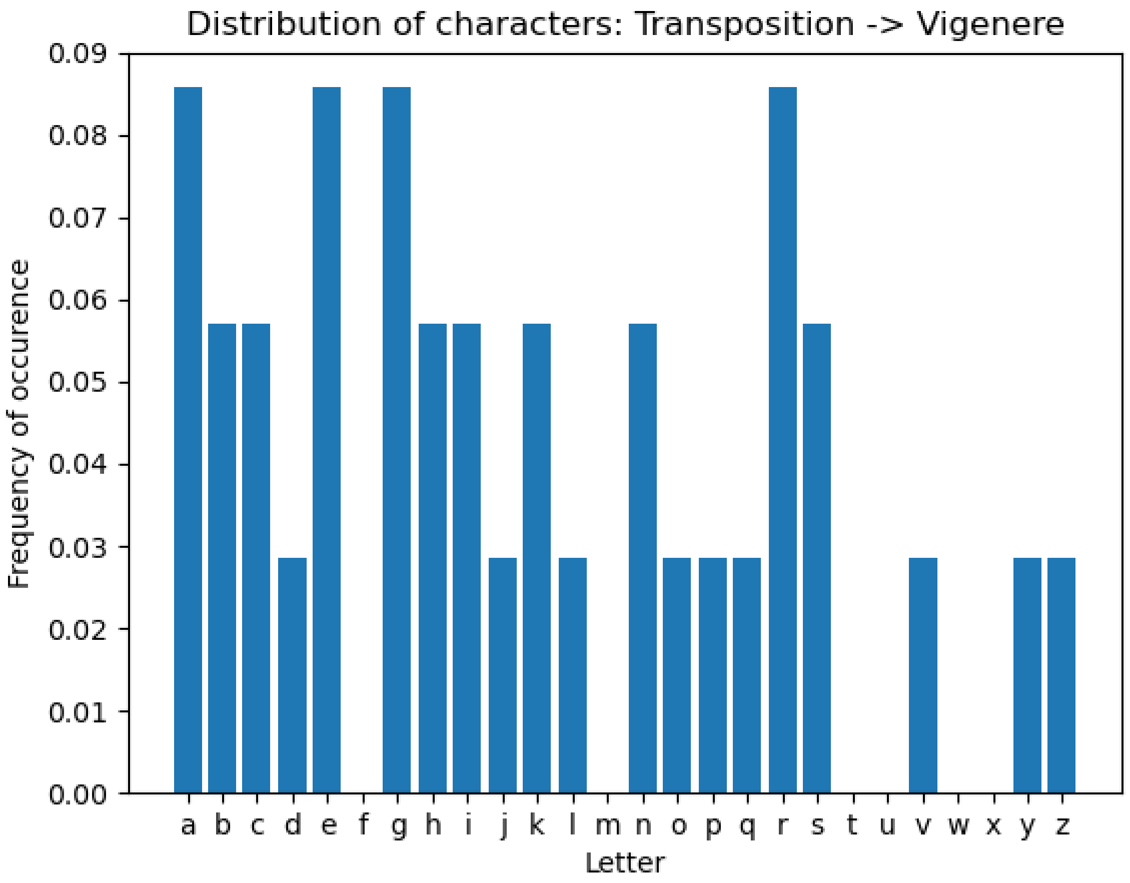
Number of combinations available: 120 Half of all possibilities to get the average time of success: 60 Delay to decipher vigenere encryption with brute-force attac k: 0.000478 seconde(s).



Row transposition cipher then Vigenere cipher

Setup: key_size_row_transposition = 5, key_size_vigenere = 3

Delay to decipher combination of encryptions with brute-force attack: 14.062653 seconde(s).



4. Discussion

To measure the strength of the encryption algorithm, I have chosen to measure the time taken for a brute-force attack. The definition of this metric is based on two assumptions: knowledge of the decryption algorithm and the size of the keys.

The test results section presents the success times of four brute-force attacks. These results is found on my own laptop and the decipher function is not optimised. It can be seen that the use of the Vigenere and row transposition algorithms alone is very low due to the small number of key combinations to be tested. The success time for Vigenere is around 0.1 seconds, compared with 0.001 seconds for row transposition. As a result, when used alone with fixed key sizes, Vigenere outperforms row transposition. With a comparable key size, this difference in performance can be explained by the much greater number of combinations with Vigenere encryption because its key uses the alphabet, whereas transposition has a key that uses the decimal base.

On the other hand, the combined use of the two methods gives much better results, with a success time of around 10 seconds. Despite this impressive progress, the brute-force attack can still be carried out in an achievable time. What's more, this metric doesn't allow us to distinguish a better condition if one exists.

To see the effect of key size, I ran the same brute-force attack test on different key sizes to see how the success time evolved. As a result of the exponential increase in combination for Vigenre encryption, we also see an exponential increase in the success time for decryption. As for the evolution for decryption with row transposition, we also see a large increase in the success time, but this increase is much smaller than that previously obtained with Vigenere encryption. These results support the fact that the size of the key determines the security of the encryption. Particular attention must therefore be paid to key generation.

I believe that the study of success time is relevant for studying the strength of an encryption method, but I think I've made assumptions that are too strong. In fact, knowledge of the decryption algorithm and key sizes is not trivial to find. It was therefore necessary to carry out an additional study to examine the strengths and weaknesses of the two methods, in particular their vulnerability to frequency analysis. The two histograms `Distribution of characters: Transposition -> Vigenere` and `Distribution of characters: Vigenere -> Transposition` show the distributions of the two ciphertext output from the encryption algorithm. We can see that the best solution is to perform the row encryption transposition first, followed by Vigenere encryption, as its distribution is close to a uniform distribution.

5. Conclusion

The aim of this study was to analyse the strengths and weaknesses of the classic transposition and substitution algorithms and their combination. Through various metrics we were able to observe the very good results of the combinations of these two methods (Vigenere and Transposition row). Thanks to a frequency analysis, we were able to define the best method with the sequence of transposition row encryption followed by Vigenere encryption. The encryption system can be improved by multiplying the layers and increasing the key sizes. To complete this study we could have defined a new metric which studies the snowball effect. We would look at the difference in output with very little change to the input. This difference could be evaluated using the Levenstein distance.