

Security Number Administration and Cryptography

Coursework 2

Niall Gallagher | 4123 Comp | 24/03/2021

**Encryption/Decryption Research**

The task assignment required me to research a suitable data encryption/decryption method and develop a prototype of a data security application that ‘The Security Company’ are seeking to use to protect their client’s confidential data. ‘The Security Company’ specifically asked that the prototype be programmed in java. The method must encrypt and decrypt text data (numbers and special characters are to be excluded) and then develop a short, written specification for your proposed solution.

I started my research looking into the most widely used ciphers around the world and doing analysis on which would be a best fit for this ‘Security Company’. My shortlist consisted of two Monoalphabetic and two Polyalphabetic ciphers. These were respectively Caesar (Monoalphabetic), Atbash (Monoalphabetic), Vigenère (Polyalphabetic) and Playfair (Polyalphabetic).

My analysis of the Caesar cipher concluded that the biggest advantage the Caesar possess is its simplicity. This main advantage makes it easier to implement an encryption method for the ‘Security Company’. It is also easy to encrypt longer text. It’s a special case of the Shift Cipher so when the key=3 it will shift the alphabet by three letters. You can also determine the length of the shift.

**Caesar**

My analysis of the Caesar cipher concluded that the biggest advantage the Caesar possess is its simplicity. This main advantage makes it easier to implement an encryption method for the ‘Security Company’. It is also easy to encrypt longer text. It’s a special case of the Shift Cipher so when the key=3 it will shift the alphabet by three letters. You can also determine the length of the shift.

But even if the opponent trying to decipher the message doesn’t know that it was encrypted using Caesar Cipher but notices that it uses some kind of Shift Cipher will figure out that there are only 26 letters. The main disadvantage of the Caesar Cipher is that it’s not a secure cryptosystem because there are only 26 possible keys to try out. An attacker can carry out an exhaustive key search with available limited computing resources. Therefore, I would not recommend this cipher for the ‘Security Company’

But even if the opponent trying to decipher the message doesn’t know that it was encrypted using Caesar Cipher but notices that it uses some kind of Shift Cipher will figure out that there are only 26 letters. The main disadvantage of the Caesar Cipher is that it’s not a secure cryptosystem because there are only 26 possible keys to try out. An attacker can carry out an exhaustive key search with available limited computing resources.

Therefore, I would not recommend this cipher for the ‘Security Company’.

**Atbash**

The next Monoalphabetic cipher I looked at was the Atbash cipher. The main advantage of this cipher is that it is quick and easy to conceal information. However, the Atbash cipher is a very weak substitution cipher as there is no secret key behind generating the ciphertext alphabet to perform the encryption. It would not take much computational power to break it therefore I would not recommend the ‘Security Company’ using this cipher.

**Playfair**

The first Polyalphabetic cipher I looked at was Playfair. It is also a substitution cipher and is difficult to break compared to the simple substitution cipher. It is a very secure cipher as there are 625 possible pairs of letters (25x25 alphabets) instead of 26 different possible alphabets. It is quick to use and requires no special equipment. The main disadvantage of this cipher is it cannot be used for the transmission of a huge amount of data. The ‘Security Company’ requires the encryption of the client’s data which would be a large amount and Playfair can only encrypt a short data set.

This is the main reason why this cipher cannot be used in my application.

**Vigenère**

The last Polyalphabetic cipher I analyzed was the Vigenère cipher. The Vigenère Cipher was designed by tweaking the standard Caesar cipher to reduce the effectiveness of cryptanalysis on the ciphertext and make a cryptosystem more robust. It is significantly more secure than a regular Caesar Cipher.

Vigenère cipher is a kind of substitution cipher that employs polyalphabetic substitutions. It is stronger than Caesar cipher as it uses series of interwoven Caesar ciphers. The cipher uses a key and a double entry table. The main disadvantage would be its repeating nature of its keys. If a cryptanalyst correctly guesses the key's length, the cipher text can be treated as interwoven [Caesar ciphers](https://en.m.wikipedia.org/wiki/Caesar_cipher), which can easily be broken individually.

**Conclusion**

Through my extensive research, I concluded that the Vigenère cipher would be best suited for the ‘Security Company’. It can process a large data amount which will be useful to protect client’s confidential data. The ‘Security Company’ can edit the encryption often by passing a new keyword into the program. The cipher will be relatively easy to implement in the Security Company’s program. The cipher being polyalphabetic will make it a lot more difficult to breakdown than a monoalphabetic cipher. I’ll draw up a robust testing plan later in this report to further back my opinion in why this cipher will be the perfect suit for this ‘Security Company’.

**Vigenère Pseudocode**

Class SecurityCompanyCipher {

public static void main (String[] args) {

String str = pass through data;

String keyword = create keyword to encrypt data;

String key = method (str, keyword);

String cipher\_text = method (str, keyword);

}

static String generateKey (String str, String Key) {

int x = str.length();

for (int i = 0;; i++)

{

if (x equals i)

i=0;

if (key.length equals str.length)

break;

key+=(key.charAt(i));

}

return key;

}

static String cipherText(String str, String key)

{

    String cipher\_text="";

    for (int i = 0; i < str.length(); i++){

int x = (str.charAt(i) + key.charAt(i)) %26;

        x += 'A';

        cipher\_text+=(char)(x);

    }

    return cipher\_text;

}

static String originalText(String cipher\_text, String key)

{

    String orig\_text="";

    for (int i = 0 ; i < cipher\_text.length() &&

                            i < key.length(); i++)

    {

        int x = (cipher\_text.charAt(i) -

                    key.charAt(i) + 26) %26;

        x += 'A';

        orig\_text+=(char)(x);

    }

    return orig\_text;

}

To Do List

Comment Pseudocode.

Add File Writer.