CS 6903 Project1

Team Members

- Jinali Sheth – N11315380
- Augustino Watson–N15292684

Introduction

Cryptanalysis refers to the study of ciphers, ciphertext, or cryptosystems (that is, to secret code systems) with a view to finding weaknesses in them that will permit retrieval of the plaintext from the ciphertext, without necessarily knowing the key or the algorithm. This is known as *breaking* the cipher, ciphertext, or cryptosystem.

Polyalphabetic Cipher

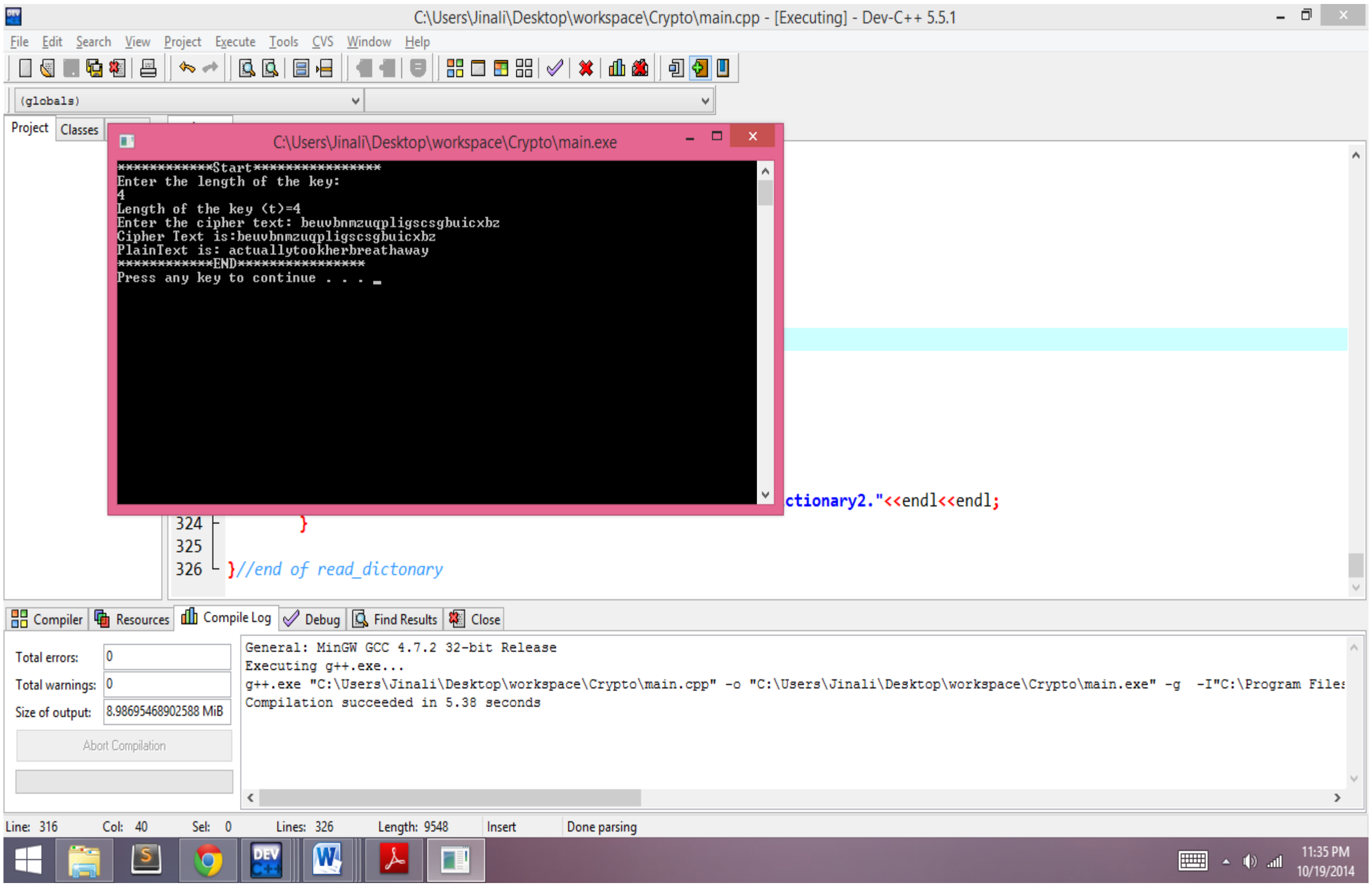
- A polyalphabetic cipher is a cipher based on substitution, using multiple substitution alphabets. The Vigenère cipher is probably the best-known example of a polyalphabetic cipher.
- Frequency Analysis is one of the methods used to break the polyalphabetic substitution cipher where we can count the frequency of cipher-text letters and then map the guessed plaintext letters to them.
- Another method called as the Kasiski Examination analysis is one of the popular methods for breaking the Vigenere ciphers where the cryptanalyst looks for repeated sequences of the cipher text and calculate the difference between those sequences.
- If the keyword length is known, the following observation of Babbage and Kasiski comes into play. If the keyword is N letters long, then every Nth letter must have been enciphered using the same letter of the keytext.
- Grouping every Nth letter together, the analyst has N "messages", each encrypted using a one-alphabet substitution, and each piece can then be attacked using frequency analysis.

Cracking Dictionary 1

For Vigenere cipher, here in this case we know the length of the key – t. Also we know that the length of plaintext will be equal to the length of ciphertext. The key is considered to be a set of repeating string of characters. Since we have the Dictionary for the plaintext, the task of determining the key gets easier.

Steps

1. Starting from the 0th index of Dictionary 1, pick up the text as long as the ciphertext length.
2. The plaintext and ciphertext are divided into substrings as long as the key. The last substring may or may not be filled completely depending on whether the ciphertext length is completely divisible by the key length.
3. Calculate the difference between each of the corresponding characters between the plaintext and the ciphertext.
4. The difference calculated is placed into a matrix having columns =t and row = substrings created.
5. If each of the rows of the matrix are equal excluding the null or zero elements, we have successfully decoded the plaintext and determined the key.
6. If not we pick up the plaintext from the dictionary starting from position 2, if previously the plaintext in Dictionary was considered from position 1.


C:\Users\Jinali\Desktop\workspace\Crypto\main.cpp - [Executing] - Dev-C++ 5.5.1
File Edit Search View Project Execute Tools CVS Window Help
(globals)
Project Classes
C:\Users\Jinali\Desktop\workspace\Crypto\main.exe
Compiler Resources Compile Log Debug Find Results Close
Abort Compilation
11:35 PM
10/19/2014

```
*************Start*****************
Enter the length of the key:
4
Length of the key (t)=4
Enter the cipher text: beuvbnmzuqpligscsgbuicxbz
Cipher Text is:beuvbnmzuqpligscsgbuicxbz
PlainText is: actuallytookherbreathaway
*************END*****************
Press any key to continue . . . _
```

```
ctionary2."<<endl<<endl;
        }
}//end of read_dictonary
```

Total errors: 0

Total warnings: 0

Size of output: 8.98695468902588 MiB

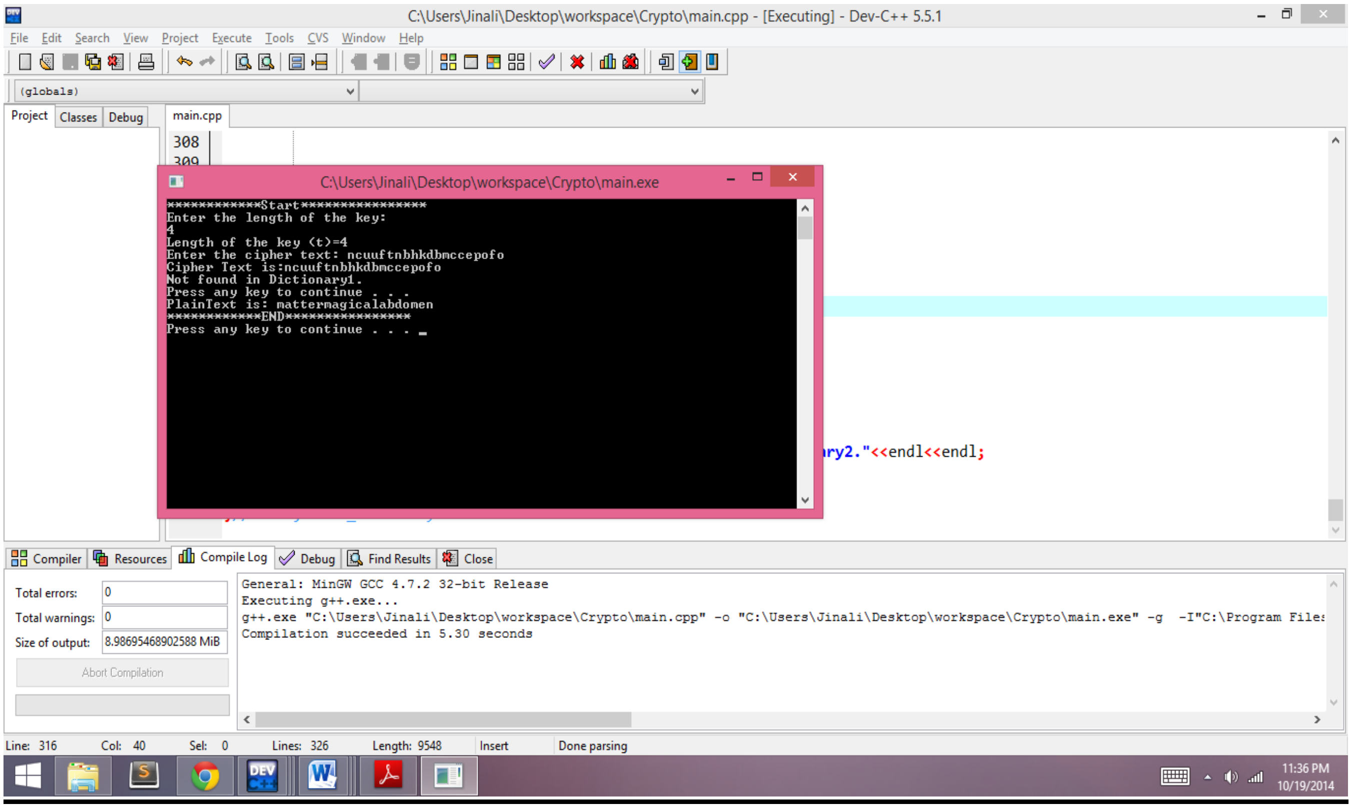
```
General: MinGW GCC 4.7.2 32-bit Release
Executing g++.exe...
g++.exe "C:\Users\Jinali\Desktop\workspace\Crypto\main.cpp" -o "C:\Users\Jinali\Desktop\workspace\Crypto\main.exe" -g  -I"C:\Program Files
Compilation succeeded in 5.38 seconds
```

Line: 316 Col: 40 Sel: 0 Lines: 326 Length: 9548 Insert Done parsing

Cracking Dictionary 2

Steps

1. Add the lengths of the words in Dictionary 2 until it adds up to the length of the ciphertext. The moment it exceeds the length of the ciphertext, discard the last word that was added.
2. Once the list of words is finalized, pass it to the same function of Dictionary 1 to decode the plaintext.
3. If the plaintext is not found, try the next combination for the same set of words.
4. When all the permutations and combinations are tried, repeat step 1 to find another possible set of words.


C:\Users\Jinali\Desktop\workspace\Crypto\main.cpp - [Executing] - Dev-C++ 5.5.1
File Edit Search View Project Execute Tools CVS Window Help
(globals)
Project Classes Debug
main.cpp
C:\Users\Jinali\Desktop\workspace\Crypto\main.exe
Compiler Resources Compile Log Debug Find Results Close
Abort Compilation
11:36 PM
10/19/2014

```
*************Start******************
Enter the length of the key:
4
Length of the key (t)=4
Enter the cipher text: ncuuftnbhkdbmccepofo
Cipher Text is:ncuuftnbhkdbmccepofo
Not found in Dictionary1.
Press any key to continue . . .
PlainText is: mattermagicalabdomen
*************END*****************
Press any key to continue . . . _
```

```
ry2."<<endl<<endl;
```

Total errors: 0

Total warnings: 0

Size of output: 8.98695468902588 MiB

```
General: MinGW GCC 4.7.2 32-bit Release
Executing g++.exe...
g++.exe "C:\Users\Jinali\Desktop\workspace\Crypto\main.cpp" -o "C:\Users\Jinali\Desktop\workspace\Crypto\main.exe" -g  -I"C:\Program Files
Compilation succeeded in 5.30 seconds
```

Line: 316 Col: 40 Sel: 0 Lines: 326 Length: 9548 Insert Done parsing

Both the team members were involved in Outlining and Testing Attack Strategies, Software Engineering and Coding, Analysis of Results, Developing test-cases, Write-Up and Final Analysis.

Extra Credit

Report of substitution ciphers

Substitution ciphers are those encryption technologies in which the cipher text is generated from the plain text by replacing the characters of the plain text characters by using a function of the input text or the input key. Substitution ciphers are generally classified into the following sub groups:

1 – Simple Substitution Ciphers

In a particularly simple implementation of substitution cipher, the message is encrypted by substituting the letter of the alphabet n placed ahead of the current letter. Consider the following simple example where n = 2

Plain Text	A	B	C	D	E	F	G	H
Cipher Text	C	E	F	G	H	I	J	K

We can decrypt this type of substitution ciphers by just moving back the characters in the cipher text by n places. The number of possible keys in such a cipher is 26, which is the key space.

2 – Homophonic substitution Cipher

In Homophonic Cipher, each letter is to be mapped to any of the 26 letter. This type of encryption also makes use of symbols and other characters which allow the letter to be mapped to more number of characters. This therefore reduces the chances of performing a frequency analysis on the cipher text. Consider the following example:

Plain Text	A	B	C	D	E	F	G	H
Cipher Text	a	d	z	f	y	l	b	e

In this the letters are randomly mapped to a different symbol so that the frequency analysis is hard to perform.

3– Polyalphabetic Cipher

In a polyalphabetic cipher multiple cipher alphabets are used. The most popular type of poly cipher is the Vigenere cipher. In the vigenere cipher, a keyword is used to encrypt the characters in the plain text. So if the keyword is 'MYKEY', the first letter of plaintext is enciphered under alphabet 'M', the second under 'Y', and so on and once exhausted, it will repeat using the symbol 'M' again. In practice, Vigenère keys were often phrases several words long. The decryption process for the vigenere cipher involves moving back the encrypted text n places according to the input given by the key. For Example:

Plain Text	A	B	C	D	E	F	G	H
Key	B	A	T	B	A	T	B	A
Cipher Text	C	C	W	F	F	X	I	I

Other variants of polyalphabetic cipher are:-

- Gronsfeld cipher
- Beaufort cipher
- Auto key cipher
- Running key cipher

4 – One time pad

The one time pad cipher or the Vernam cipher is a provable secure cipher. This cipher makes use of a long bit string key as a key for encryption. The key has to be the same length as the message to be encrypted. The key is then XOR ed with the plain text to get the cipher text. Another way to say this is that we add key bits with the plain text modulo 2 to get the cipher text. For example:

Plain Text	H	I	T	L	E	R
Key	111	101	110	101	111	100
Cipher Text	110	101	100	001	110	111
Letter	S	R	L	H	S	S

Now, for decryption we transmit the encrypted text and then decrypt using the same key.

Plain Text	S	R	L	H	S	S
Cipher Text	H	I	T	L	E	R

Cryptanalysis approaches for substitution ciphers

Several approaches have been suggested to break the substitution ciphers, each having its own pros and cons. Substitution ciphers being the foundation since the time of classical cryptography has been through enough analysis on the basis of which the complex cipher techniques have been built. In order to optimize our approach for breaking the cipher text it is important to exploit the vulnerabilities that substitution ciphers have. Many of the following approaches are thus based on the same.

The main goal of the attack is to determine the key which is used for encryption. Once the key is not secret anymore, messages are as good as decrypted. Different approaches have been analyzed in theory to find the key of the cipher and by decrypting the whole cipher text.

Different approaches to break the substitution ciphers include the following:

1. Exhaustive search.
2. Simulated annealing.
3. Frequency analysis.
4. Genetic algorithm.
5. Particle swarm optimization.
6. Tabu search
7. Relaxation algorithm.

Each of them can be described as follows.

1. Exhaustive search.

It is the simplest of all algorithms to break the substitution cipher. This approach can be used when there is a finite key space and allows all possible key combinations to be checked until the correct one is found. This method can be used for breaking a mono alphabetic shift cipher. While the time consumed is the highest, results are accurate.

2. Frequency analysis.

Frequency analysis is based on the fact that, in any given stretch of written language, certain letters and combinations of letters occur with varying frequencies. Moreover, there is a characteristic distribution of letters that is roughly the same for almost all samples of that language. Thus, we can use it to first count the frequency of the letters in cipher-text and then map it to the guessed letters in plaintext.

3. Genetic algorithm.

Genetic algorithms are a population based meta heuristics. They have been applied to many optimization problems. Basically it is used for encrypting real time data transmission. It generates pseudorandom sequence using non-linear feed-back shift register. The feedback shift register is a mechanism for generating extremely well binary sequence. Second we use the generated pseudorandom sequence with crossover operator for encrypting the data. This is partially used is it does not produce the exact result. Earlier, the whole key space with all possible substitution ciphers were searched, followed by the modification of searching the more likely used generated keyspace.

4. Particle swarm optimization.

It is a computational method that optimizes a problem by iteratively trying to improve a candidate solution with regard to a given measure of quality. It uses a binary representation to attack the Merkle-Heleman knapsack algorithm. Cryptanalysis starts from cipher text, which is in the integer form. Each number represents as a target sum of the hard knapsack problem. The final goal of this algorithm is to translate each number in the correct knapsack algorithm which will represent the ASCII code for the plaintext characters.

5. Tabu search

The tabu search prevents the search from returning to a previously explored region of the solution space too quickly. This is achieved by retaining a list of possible solutions that have been previously encountered. These solutions are called 'tabu'; hence the name of the technique. The size of the tabu list influences the performance of the algorithm. Tabu search is similar to simulated annealing with the added constraint of the tabu list. Two randomly chosen key elements are swapped to generate candidate solutions. In each iteration, the best new key formed replaces the worst existing one in the tabu list.

6. Relaxation algorithm.

Probabilistic relaxation algorithms are iterative parallel classification algorithms. The process of an element in a graph structure trying to estimate its class membership probabilities based on those of its neighbors is iterated until a satisfactory classification is achieved. Relaxation is a problem solving technique for constraint-satisfaction problems, such as those found in image and image processing.

7. Simulated annealing.

It is similar to the Hill Climbing algorithm. It tries finding the global optimum avoiding into thinking the local optimum as the best solution. The worse solutions are chosen over better ones according to probability.