DAT 510 Assignment 1)

Abstract

The focus of this project was to solve problems related to implementing and cracking cryptographic ciphers. The project consisted of two main parts, the first of which was related to polyalphabetic ciphers. In this part the main task consisted of developing tools to analyze and crack encrypted text for which the specific method and key from which the encrypted text was produced, was unknown. This task was executed successfully, where the cipher method was identified as an *autokey-cipher,*with the key being found as: “DATFBA”. The second part consisted of implementing and utilizing simplified DES, a lightweight model-version of the larger DES (data encryption standard), on which we were to encode and decode binary strings. Followingly, TripleSDES a threefold composition of DES ciphers, was also implemented. From these implementations a part of the task was to find the key on which two binary messages was encoded on each respective algorithm. Using a brute force approach, the key was found to be “11111010” for the SDES encoded message and “11111010--0101011111” for the TripleSDES encoded message.

Introduction

The field of cryptography is vital for the modern data and communication technology, being the primary source of means for ensuring secure communication, privacy, and data security. To get some semblance over how methods from this field is utilized in the real word, this project consists of solving simplified but relevant problems. More specifically, these problems consist of implementing and cracking different cipher algorithms.

Design and Implementation

Part 1

Task 1.

Of all the tasks in the project, this task proved to be the most challenging and time consuming. At first an attempt was made to decode the text as a Vigenère cipher, but this yielded no intelligible results. After this attempt, an implementation for the autokey cipher algorithm yielded the correct result. The general method for performing cryptanalysis on such a cipher is described on Wikipedia [1]. Hence the task was to create the necessary tools to be able to perform this analysis. The solution can be summarized by first describing the general method, and then describe the specific implementation.

An autokey substitution cipher is a polyalphabetic substitution-cipher which is like the Vigenère cipher in that it consists of a plaintext and a key of equal length, where the characters in both texts belong to a given alphabet. The characters in the alphabet can then be treated as numeric values, and the result of the cipher is the sum of the key and plaintext value. Where autokey differs from Vigenère is when how the key may be generated. In the case of Vigenère the key may be a simple keyword repeated throughout the length of the plaintext. With this type of encoding, statistical characteristics of the language is preserved for every key-length spaced letter, and cryptanalysis can be performed by bucketing the text and analyzing the statistical similarity to English text. The autokey cipher works by appending a keyword to the front of the plaintext and using this as the key. The advantage of this method is that the statistical properties of the text is not preserved, and the statistical methods one would use in the Vigenère case will not work. Rather, the strategy for cracking an autokey cipher can be inducted by first realizing that the ciphertext itself is most of the key in the cipher. If one were to guess or know a word of length n which appears in the plaintext, one would also know that this word encodes the next n letters a keyword-length further down the plaintext. Furthermore, one would also be able to find the word which together with the ciphertext n letters up, would produce the guessed/known word. From this fact, one can lead the following strategy: Choose a word/n-gram that is likely to appear in the plaintext such as “the, and, tha, …” then for key-lengths up to a set limit, decode the next and previous n-gram one key length apart. If the n-grams that results from the decoding seems likely to appear in the plaintext one can repeat the process. Should one guess correctly, then it is likely that enough of the plaintext is revealed to be able to surmise the contents.

The main component of the solution is the python class Autokey\_cryptanalysis, which is a class which stores the given ciphertext and methods to process and develop this text into the sought solution.

[1] https://en.wikipedia.org/wiki/Autokey\_cipher