

DAT 510: Assignment 1

Submission Deadline: 23:59, Monday, Sept. 20, 2021

Cryptanalysis of primitive ciphers

In this assignment, you will try your skills at cracking some encrypted messages.

Warning: Although the encryption techniques used in this assignment are extremely primitive compared to practical encryption schemes used in the real world, they are not necessarily easy to solve (even with computer assistance). Start early and deadline for submission is soon!

Part I. Poly-alphabetic Ciphers

For this part of the assignment, you are given enciphered English text and a hint about the encryption algorithm that was used. Your mission: Develop the necessary (software) tools and use them to help you produce plaintext.

Task 1 The text was enciphered using a polyalphabetic substitution cipher, where the key length was no larger than 10. Blank spaces were first deleted and then inserted at convenient locations.

FRRUU OIIYE AMIRN QLQVR BOKGK NSNQQ IUTTY
IIEEA WIJTG LVILA ZWZKT ZCJQH IFNYI WQZXH
RWZQW OHUTI KWNNQ YDLKA EOTUV XELMT SOSIX
JSKPR BUXTI TBUXV BLNSX FJKNC HBLUK PDGUI
IYEAM OJCXW FMJVM MAXYT XFLOL RRLAA JZAXT
YYWFY NBIVH VYQIO SLPXH ZGYLH WGFSX LPSND
UKVTR XPKSS VKOWM QKVCR TUUPR WQMMY XTYLQ
XXYTR TJJGO OLMXV CPPSL KBSEI PMEGC RWZRI
YDBG E BTMFP ZXVMF MGPVO OKZXX IGGFE SIBRX
SEWTY OOOKS PKYFC ZIEYF DAXKG ARBIW KFWUA
SLGLF NMIVH VVPTY IJNSX FJKNC HBLUK PDGUI
IYEAM HVFDY CULJS EHHMX LRXBN OLVMR

Hints In the ciphertext the letters were first converted to upper case—thus the alphabet substitutions consist of permutations of the 26 upper case letters A through Z. Spaces were removed before encryption and reinserted after encryption. Suppose that substitution 1 maps A to X, and substitution 2 maps B to Y. The plaintext message “AB ABAB ABA” (with spaces) might be converted to the ciphertext “XYXYX YXYX”. You might consider

using the statistical analysis techniques discussed in class to crack these problems.

Task 2. Try shorter and Longer key lengths in your program and use the time package to find out program execution time. Does the program take longer time to decrypt the ciphertext? how does execution time arise by adding letters to the key?

Hints: Here are useful links about how to measure execution time in Python And Javascript, You can also find the function for your preferred programming language using search engines.

Task 3. This ciphertext has been encrypted with the same Key as previous ciphertext(Task1) with an addition in the encryption process. Is it possible to Produce the plaintext again using the same tool you created in Task1? Explain the differences you have encountered.

IRKZV ONZPY UAQZL ULCDI OEVWF ETBAW SHLGO
YQSXT UQRRK LRQUT FHUSE ZBFPR BEPHY DYEKF
ZSPPT VYQSY GKUHJ GNHXN UMWFF XIZFN NLWTJ
CKYHZ YDPDX KCOUO JEOMU AKVAU EGUEX RKHFC
SNHGG WRABW RASXJ IFJHO JRLLJ KOQLO UQRIT
YHVFV GZGRM TLRQJ ZGNNP NYJAE DFLQI SLYSV
RVKLE AJUNL MHDGE IFFQN FKEKT NJGQN OPOXM
VVRRC JGHEH FEVGB QDAEI FDHTA AWFYG ZLLVO
AUXFV JRPGV DYOYK BFMQA TWFM S WUQEB PQHXC
WWEUP LGSGL NYMIM RXOWK FZFOE FUBFG QFNVI
OVLHZ NETBS AIBBT PEIHQ DRTAU EGUEX RKHFC
SNHGG PDDHY OBG OV CJBXG DVEIZ LWMJS

Hints: Here same encryption mechanism has been used with same key. ”**with an addition in the encryption process**”.

Part II. Simplified DES

In this section, you will implement a simplified version of the DES block cipher algorithm. Naturally enough, it is called SDES, and it is designed to have the features of the DES algorithm but scaled down so it is more tractable to understand. (Note however, that SDES is in no way secure and should not be used for serious cryptographic applications.)

The photocopied handouts that accompany this project description give the detailed specifications of SDES. SDES encryption takes a 10 bit raw key (from which two 8 bit keys are generated as described in the handout) and encrypts an 8 bit plaintext to produce an 8 bit ciphertext.

To verify that your implementation of SDES is correct, try the following test cases:

| Raw Key | Plaintext | Ciphertext |
|------------|-----------|------------|
| 0000000000 | 10101010 | 00010001 |
| 1110001110 | 10101010 | 11001010 |
| 1110001110 | 01010101 | 01110000 |
| 1111111111 | 10101010 | 00000100 |

Task 1. Use your implementation to complete the following table:

| Raw Key | Plaintext | Ciphertext |
|------------|-----------|------------|
| 0000000000 | 00000000 | ? |
| 0000011111 | 11111111 | ? |
| 0010011111 | 11111100 | ? |
| 0010011111 | 10100101 | ? |
| 1111111111 | ? | 00001111 |
| 0000011111 | ? | 01000011 |
| 1000101110 | ? | 00011100 |
| 1000101110 | ? | 11000010 |

The DES algorithm uses keys of length 56 bits, which, when DES was originally designed, was thought to be secure enough to meet most needs. However, due to Moore's law, the increase in computing power makes it more tractable to brute-force crack a 56-bit key. Thus, an alternative version of DES using longer keys was desirable. The result, known as Triple DES uses two 56-bit raw keys k_1 and k_2 and is implemented by composing DES with itself three times in the following way:

$$Enc_{3DES}(p) = Enc_{DES}(k_1, Dec_{DES}(k_2, Enc_{DES}(k_1, p))) \quad (1)$$

Here, p is the plaintext to encrypt, Enc_{DES} is the usual DES encryption algorithm and Dec_{DES} is the DES decryption algorithm. This strategy doubles the number of bits in the key, at the expense of performing three times as many calculations.

The TripleDES decryption algorithm is just the reverse:

$$Dec_{3DES}(c) = Dec_{DES}(k_1, Enc_{DES}(k_2, Dec_{DES}(k_1, c))) \quad (2)$$

Task 2. Implement a class called TripleSDES and complete the following table

| Raw Key 1 | Raw Key 2 | Plaintext | Ciphertext |
|------------|------------|-----------|------------|
| 1000101110 | 0110101110 | 11010111 | ? |
| 1000101110 | 0110101110 | 10101010 | ? |
| 1111111111 | 1111111111 | 00000000 | ? |
| 0000000000 | 0000000000 | 01010010 | ? |
| 1000101110 | 0110101110 | ? | 11100110 |
| 1011101111 | 0110101110 | ? | 01010000 |
| 1111111111 | 1111111111 | ? | 00000100 |
| 0000000000 | 0000000000 | ? | 11110000 |

Task 3. Cracking SDES and TripleSDES

- The message in the file cxt1.txt was encoded using SDES. Decrypt it, and find the 10-bit raw key used for its encryption.
- The message in the file cxt2.txt was encoded using TripleSDES. Decrypt it, and find the two 10-bit raw keys used for its encryption.

Hints. The ciphertexts are obtained by encrypting the binary string converted from clear message with the standard ASCII code.

Task 4. Webserver using TripleSDES

Create a simple webserver which already stores two Raw keys (1000101110 -0110101110) and can decrypt any ciphertext coming to it by that key using TripleSDES which you have already created and show the result in the browser. This is a simple end to end encryption. How strong is the security in this type of communication?

Example:

Browser input(bits are just for demo):

`http://localhost:5000/index.js?cipher=101101101011101111100101111011100010111`

Output: Hello

Hints. here are useful links on how to make simple webserver using Python(Flask)and Javascript (Nodejs)

Assignment Submission

Deadline: 23:59, Monday, Sept. 20, 2021 (submit your assignment through canvas)

Final submission:

1. Source Code

- Source code submitted for the assignment should be your own code. If you have used sources from the internet everything should be added to the references. If you used someone's code without reference, that will also be treated as plagiarism.
- Source code should be single, compressed directory in .tar.gz or .zip format.
- Directory should contain a file called README that describes the contents of the directory and any special instructions needed to run your programs (i.e. if it requires and packages, commands to install the package. describe any command-line arguments with the required parameters).
- You may use any “reasonable” programming language for part one of the assignment. Reasonable languages include: Java, C, C++, Python, MatLab, R and others with permission of Jayachander Surbiryala (Email: jayachander.surbiryala@uis.no) or Ravishankar Bhaskarrao Borgaonkar (Email: ravishankar.b.borgaonkar@uis.no).
- You should **NOT** use available libraries/packages/classes for implementing the core functionality of the assignment.

2. A **separate** report with PDF format

- Texts in the report should be readable by human, and recognizable by machine;
- Other formats will **NOT** be opened, read, and will be considered missing;
- Report should follow the formal report style guide in next page.
- Each student should write an individual report. Each report will be checked for plagiarism. If it is copied from some where else, you will fail the assignment.

3. Upload a presentation (with audio)

- Prepare slides, present what you have done.
- Execute the code and show the results.
- Explain the results (In audio/Video)
- This can be done using PPT (you can use other tools, if you are comfortable)

NOTE: If you encounter problem with upload archive file (e.g. *.zip, *.tar) to the website <https://stavanger.instructure.com/>, you should be able to upload after you add extension .txt to your archive file (e.g., *.tar ⇒ *.tar.txt).

Note: The assignment is individual and can NOT be solved in groups.

Project Title

Abstract

A one-paragraph summary of the entire assignment - your procedure, results, and analysis.

Introduction

Describe the background for the project. If you are building on top of any existing resources highlight them in this section and cite them in your references.

Design and Implementation

Detail description of the design, procedure, and implementation of your project along with the following details from Part I and Part II.

Part I

- The plaintext message you managed to decipher;
- Describe the strategy you employed, show the details for each of the steps of that strategy, describe any programs you wrote, show sample output of these programs, and show how you transformed that output into your solution.
- Describe the Execution time and impact of the key length on it.

Part II

- The result of test cases in **Tasks 1 and 2**;
- The bits making up the keys of the SDES and TripleDES in **Task 3**;
- Describe the filtering strategy you used to know that the keys are correct.
- Describe **Task 4** implementation and how strong is the security in this type of communication?

Test Results

Results of testing the software, as you observed/recorded them. Note that this section is only for observations you make during testing. Your analysis belongs in the Discussion section.

Discussion

Your analysis of what your testing results mean, and your error analysis.

Conclusion

A short paragraph that restates the objective from your introduction and relates it to your results and discussion and describes any future improvements on your techniques that you would recommend.

Works Cited

A bibliography of all of the sources you got information from in your report.

CTX1.txt

0100011100000000101000000110011011100101100000001011101000000000101101110010101110101
011101101110010001110000000101000111101110100100111110001000010001110110111001001100
101011111001011101101110011011101011101001001111101011110000100101001010100010000100
11111100110110010111010011110011001000000000101010111011011110100100000100111110101111
010001111010111101110100011101000000000101001100000000010110111010111010100010000100
011101101110010011001010111110010111000000011000100010010000

CTX2.txt

000000011010011100110010110001100110010010100111110101111010011110011100011101000111
0100100111000000000011010011100000001100110011010000111011010000000011001110011101111
011111100010010010011100100111001001100110100001011111101010000010110011110110101010
000111000110001001001010000100100011101001110111010010011100010000011010000101111110
000000010111111011010111110101111010011111101111101001111001110010011001110110100000
000110011100111011110111111000100100101001111101101001000001