DAT 510: Assignment 1

Submission Deadline: 11:59 pm, Tuesday, Sept. 22, 2020

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.

Task1. This 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.

BQZRMQ KLBOXE WCCEFL DKRYYL BVEHIZ NYJQEE BDYFJO PTLOEM EHOMIC UYHHTS GKNJFG EHIMK NIHCTI HVRIHA RSMGQT RQCSXX CSWTNK PTMNSW AMXVCY WEOGSR FFUEEB DKQLQZ WRKUCO FTPLOT GOJZRI XEPZSE ISXTCT WZRMXI RIHALE SPRFAE FVYORI HNITRG PUHITM CFCDLA HIBKLH RCDIMT WQWTOR DJCNDY YWMJCN HDUWOF DPUPNG BANULZ NGYPQU LEUXOV FFDCEE YHQUXO YOXQUO DDCVIR RPJCAT RAQVFS AWMJCN HTSOXQ UODDAG BANURR REZJGD VJSXOO MSDNIT RGPUHN HRSSSF VFSINH MSGPCM ZJCSLY GEWGQT DREASV FPXEAR IMLPZW EHQGMG WSEIXE GQKPRM XIBFWL IPCHYM OTNXYV FFDCEE YHASBA TEXCJZ VTSGBA NUDYAP IUGTLD WLKVRI HWACZG PTRYCE VNQCUP AOSPEU KPCSNG RIHLRI KUMGFC YTDQES DAHCKP BDUJPX KPYMBD IWDQEF WSEVKT CDDWLI NEPZSE OPYIW

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.

Task2. 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.

Task3. This ciphertext has been encrypted by 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.

BQZRMQ KLAYAV AYITET EOFGWT EALRRD HNIFML BIHHQY XXEXYV LPHFLW UOJILE GSDLKH BZGCTA LHKAIZ BIOIGK SZXLZS UTCPZW JHNPUS MSDITN OSKSJI EOKVIL BKMSZB XZOEHA KTAWXP WLUEJM AIWGLR TZLVHZ SATVQI HZWAXX ZXDCIV TMLBIQ RWZMLB VNGVQK AIZBXZ HVVMMA MJLRIW GKITZL VHZRRV YCBTVM FVOIYE FSKGKJ AVWHUV BUHZSA EFLHMQ HHVSGZ XIKYTS YZXUUC KBTOGU VABLDP BGJCGF NLIIYA HJFWGG PSCPVA ZEASME MLGOYR CGFXVG EJTTTW TSAAIL QFKEEP CPULXW WZRLVI VVYUMS MSILRI IBLWJI TKWUXZ GUZEJG DUCQEE QEOBTP SIHTGW UALVMA ILTAEZ TFLDPE IVEGYH PLZRTC YJVYGX ABFNPQ XLCEYA RGIFCC WHBGIF WSYLBZ MDWFPX KZSYCY APJTFR CKTYYU YICYLR ZALETS DWHMGR PTTGUW CGFNTB JTRNWR AADNPQ XLTBGP RZMJTF KGTSPV DTVAPE ZPRIP

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 Moores law, the increase in computing power makes it more tractible 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 k1 and k2 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 perfoming 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 mesage 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.

Task4. 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=10110110101110111111001011110111000101111

Output : Hello

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

Assignment Submission

Deadline: 11:59 pm, Sept. 22, 2020 (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, Javascript, R, Go and others with permission of **Sohrab Chalish** (s.chalishhafshejani@stud.uis.no) /**Dhanya Therese Jose** (dhanya.t.jose@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 plagarism. If it is copied from some where else, you will fail the assignment.

3. A presentation on the powerpoint including your voiceover describing the steps and challenges during implementation of each part. This should not be more than 10 minutes and you can include parts of your code and explain what you have done.

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

Part I

- The plaintext message you managed to deciper;
- 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.

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

01000111000000001010000001100110111001011000000010111010000000001011011100101011101
01111011011100100011100000001010001111011101001001111100010000100011101101110010011
00101011111001011101101110011011101011101001001111101011110000100101001010100010000
10011111100110110010111010011110011001000000001010101110110111010010000010011111010
11110100011110101111011101000111010000000001010011000000000101101110101110101000100
00100011101101110010011001010111110010111000000011000100010010000

CTX2.txt

00000001101001110011001011000110011001001010011111010111101001111001110001110100011
10100100111000000000110100111000000011001100110100001110110100000000110011100111011
11011111100010010010011100100111001001100110100001011111101010000010110011110110101
01000011100011000100100101000010010001110100111011101001001110001000001101000010111
11100000000101111110110101111101011110100111111011111010011110011100100110011101101
00000000110011100111011110111111000100100101001111101101001000001