# C&O 487/687: Assignment #1

Due date: September 22, 2022 (Thursday)

Note: Some questions use randomization to customize to you specifically. Please include your max-8-character UW user id (`j2tham`) at the beginning of your answer so we can look up your custom solution.

1. [13 marks] **Cryptanalysis of historical ciphers**

   Please include your max-8-character UW user id (`j2tham`) at the beginning of your answer so we can look up your custom solution.

   For this question, you need to obtain the ciphertexts personalized to you. Download the file `https://www.math.uwaterloo.ca/~dstebila/as/as.cgi/download/co487_f22/a1q1ciphertexts.zip`.

   Your folder contains 3 ciphertexts, each of which is an encryption of some English plaintext written in a similar style of text. The plaintext may start in the middle of a sentence and may end in the middle of a word.

   Each plaintext is different and each is encrypted with a different cipher. The three ciphers used, in random order in your set, are:

   - random simple substitution cipher
   - Vigenère cipher
   - transposition cipher

   All plaintexts are written in English and use upper and lower case letters. For the substitution and Vigenère ciphers, punctuation, spaces, and numbers are not encrypted, and should be ignored. For the transposition cipher, all characters are subject to transposition: letters, numbers, punctuation, spaces.

   Your task is to determine which cipher was used for each ciphertext, and what the corresponding plaintext is, using the following steps. You may use online tools to assist, such as CrypTool (`https://www.cryptool.org/en`), but please indicate which tool(s) you used. If you write your own software to help you solve, please include that in your submission.

   (a) [3 marks] For each ciphertext, using either a table or a histogram, present the following characteristics:

   - the single character frequencies; and
   - the 20 most common digram frequencies. (i.e., which pairs of letters appear most).

   Include, for *each ciphertext*, a description of the relevant properties of *each characteristic*. In particular how uniform are the frequencies. In addition to providing relevant tables and histograms, it is recommended that you answer this question by providing a 2 by 3 grid, characteristics on one axis and ciphertexts on the other, with a brief comment in each square.

   (b) [4 marks] Explain clearly how each of the two characteristics measured in part (a) is expected to look for the ciphertext from each of the three cipher algorithms used. It is recommended that you answer this question by providing a 2 by 3 grid, characteristics on one axis and ciphers algorithms on the other, with a brief comment in each square.

   Use the measured values to argue which ciphertext comes from which of the three ciphers.

   (c) [3 marks] Explain, with direct reference to the statistics that you found, the procedure you can use to cryptanalyze each cipher. You are not expected to perform the cryptanalysis in this part – you need to explain how it can be done in principle for each type of cipher and how the measured statistics can help.

(d) [3 marks] Obtain the plaintext. Indicate what the key was in each case. (You can use automatic tools like CrypTool where applicable.) It is okay if the last few characters of your plaintext are nonsensical, as the plaintext may have been interrupted in the middle of a word and may not be a multiple of the block length of the Vigenère or transposition cipher.

2. [7 marks] **Cryptanalysis of an LFSR**

Please include your max-8-character UW user id (`j2tham`) at the beginning of your answer so we can look up your custom solution.

In the lectures on stream ciphers, we noted that stream ciphers do not use the output of linear feedback shift registers (LFSRs) directly, instead they apply some non-linear filter or combiner prior to generating the output. In this question, we will see why, by developing a technique to cryptanalyze an LFSR with unknown feedback coefficients using a known-plaintext attack.

Let $(k_0, \dots, k_{m-1})$ be the $m$-bit key that is used to initialize the LFSR, in other words,

$$s_0 = k_0, s_1 = k_1, \dots, s_{m-1} = k_{m-1} \enspace .$$

Let the LFSR be defined by the recurrence relation

$$s_{i+m} = \sum_{j=0}^{m-1} c_j s_{i+j} \bmod 2$$

for $i \geq 0$, where $c_0, \dots, c_{m-1} \in \mathbb{Z}_2$.

Suppose we know an $n$-bit plaintext $x_0, x_1, \dots, x_{n-1}$ and the corresponding ciphertext $y_0, y_1, \dots, y_{n-1}$, in other words,

$$y_i = x_i \oplus s_i \tag{1}$$

for $i \geq 0$.

If $n \geq 2m$, we can derive a system of $\geq m$ linear equations in $m$ unknowns, which can then be solved.

Suppose Eve obtains the ciphertext

0111111110000101

corresponding to the plaintext string

1000111010110000

and suppose we know that this was generated a 4-stage LFSR with unknown feedback coefficients, namely

$$s_{i+4} = c_0 s_i + c_1 s_{i+1} + c_2 s_{i+2} + c_3 s_{i+3}$$

(a) [1 marks] Compute the keystream used to encrypt.

(b) [1 marks] What is the initial state of the LFSR?

(c) [2 marks] Write a matrix equation for the unknown feedback coefficients $c_0, c_1, c_2, c_3$ in terms of the keystream.

(d) [2 marks] Solve the linear system for the feedback coefficients.[1]

(e) [1 marks] Draw LFSR with the initial state filled in.

[1] This calculator may be helpful for computing matrix inverses modulo 2: `https://planetcalc.com/3324/`

3. [7 marks] **Keystream re-use**

   Due to his eclectic interests which range from young-adult fantasy literature to rock music, Bob has amassed quite the collection of books which he has archived digitally. Since Bob is taking CO 487 he decides to flex his newfound cryptography muscles and encrypt his digital archive.

   For the purpose of archiving, Bob first converted his book collection to plain text and then pre-processed them by capitalising Roman letter characters and replacing non-letter characters by an underscore, ‘_’, and then collapsing multiple underscores to a single underscore. For example, Bob would pre-process the text ‘`Beam me up, Scotty!!`’ to ‘`BEAM_ME_UP_SCOTTY_`’. Consequently, pre-processed text consists of characters from the alphabet

   $$\Sigma = \{x \mid x \text{ is an uppercase Roman letter}\} \cup \{\_\} \ .$$

   To encrypt the pre-processed text, Bob uses a stream cipher which produces keystreams consisting entirely of characters from the alphabet $\Sigma$. Fixing a keystream $k$, encryption and decryption are defined as

   $$\mathsf{Enc}_k : \Sigma^{|k|} \to \Sigma^{|k|} \quad m \mapsto \Big\|_{i=1}^{|k|} f^{-1}\big(f(m_i) \oplus f(k_i)\big)$$

   $$\mathsf{Dec}_k : \Sigma^{|k|} \to \Sigma^{|k|} \quad c \mapsto \Big\|_{i=1}^{|k|} f^{-1}\big(f(c_i) \ominus f(k_i)\big)$$

   where $f : \Sigma \to \mathbb{Z}_{27}$ is the bijection

   $$\begin{gathered} \texttt{\_} \xleftrightarrow{f} 0 \\ \texttt{A} \xleftrightarrow{f} 1 \\ \vdots \\ \texttt{Z} \xleftrightarrow{f} 26 \end{gathered}$$

   and $\oplus$ ($\ominus$) denotes addition (subtraction) in $\mathbb{Z}_{27}$ and $\|_{i=1}^{n} x_i$ denotes the concatenation $x_1 \| x_2 \| \ldots \| x_n$.

   Being only a novice cryptographer, Bob makes a fatal mistake — he fails to initialize the encryption scheme's keystream generator with a fresh IV for each plaintext.

   (a) [2 marks] Implement the functions Enc and Dec in a programming language of your choice.

   (b) [2 marks] Available on LEARN are two ciphertext files `q3_c1.txt` and `q3_c2.txt` which contain ciphertexts $c_1$ and $c_2$ respectively. Does $m_1$, the plaintext corresponding to $c_1$, include the text ‘`LOSE_OUR_HEADS`’? Justify your answer.

   (c) [3 marks] Fully decrypt the ciphertexts $c_1$ and $c_2$ and identify their original source. Explain the procedure you used.

4. [5 marks] **Double encryption**

   Let $(E, D)$ be a symmetric key encryption scheme. Consider the “double encryption” symmetric key encryption scheme consisting of the algorithms $(E', D')$ where

   $$E'(k, m) = E(k, E(k, m)) \qquad D'(k, c) = D(k, D(k, c))$$

   We might naively expect that, if encrypting a message once with $E$ is good, encrypting a message twice with $E$ is even better! However, this is not always true.

   (a) [2 marks] Show that there is a symmetric key encryption scheme $(E, D)$ such that double encryption is not secure. The scheme you rely on should be at least semantically secure against a ciphertext-only attack by a computationally bounded adversary.

(b) [3 marks] Prove that if $(E, D)$ is a secure symmetric key encryption scheme (i.e., semantically secure against a chosen-plaintext attacks by a computationally bounded adversary) then double encryption is also secure.

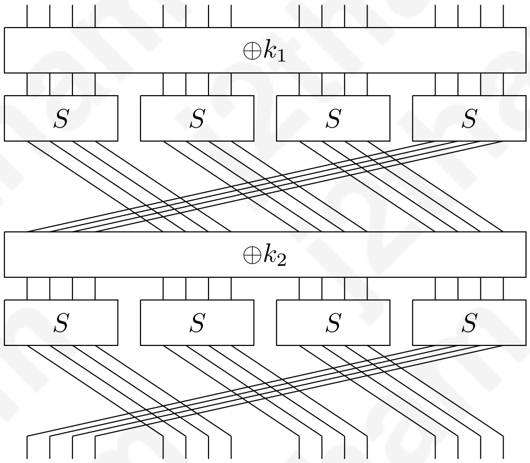
You can do this proof using the contrapositive. In other words, to show that "if $(E, D)$ is secure, then double encryption is secure", we could equivalently show "if there exists an adversary SuperEve that can break double encryption, then there exists an adversary Eve that can break $(E, D)$."

Here's the approach.

Imagine that Eve's task is to break $(E, D)$ – in other words, to learn some partial information about the plaintext $m^*$ behind a ciphertext $c^* = E(k, m^*)$ using a chosen-plaintext attack, meaning Eve can input any message $m'$ and get back $c' = E(k, m')$. Furthermore, imagine Eve has a friend SuperEve who does know how to break the double encryption $(E', D')$ – in other words, if SuperEve is given a ciphertext $\hat{c} = E(k, E(k, \hat{m}))$, and can get answers to chosen plaintext queries, namely SuperEve can input any message $\hat{m}'$ and get back $\hat{c}' = E(k, E(k, \hat{m}'))$, then SuperEve can output some partial information about the plaintext $\hat{m}$. Explain how Eve can make use of SuperEve to learn some partial information about $m^*$, such that Eve's runtime is small (excluding SuperEve's runtime). In other words, explain how Eve can set things up so that SuperEve effectively solves the problem for Eve.

5. [7 marks] **Substitution-permutation networks**

Suppose we have a block cipher constructed using the following substitution-permutation network:



It takes as input a 16-bit message and a key, and outputs a 16-bit ciphertext. It expands the 16-bit key into two 16-bit round keys $k_1$ and $k_2$. It has two rounds. In each round, the same 4-bit $S$-box is used on each chunk of 4 bits, and the permutation maps bit $i$ to bit $i + 4 \bmod 16$.

(a) [1 mark] What is the complexity of a generic brute force attack on the keyspace?

(b) [2 marks] Explain how you could carry a chosen plaintext attack with much lower complexity that would totally break the cipher. In particular, show a way to be able to decrypt any ciphertext using only 16 chosen plaintext queries.

(c) [1 mark] Would the cipher be more secure if it had 16 rounds instead of 2 rounds? Why or why not?

(d) [1 mark] Would the cipher be more secure if it used a different $S$-boxes in each row? Why or why not?

(e) [1 mark] Would the cipher be more secure if it had 128-bit keys and operated on 128-bit blocks (with 32 applications of the same 4-bit $S$-box in each round, and the obvious extension of the permutation)? Why or why not?

(f) [1 mark] What is the fundamental flaw in this design? How could you fix it?

## Academic integrity rules

You should make an effort to solve all the problems on your own. You are also welcome to collaborate on assignments with other students in this course. However, solutions must be written up by yourself. If you do collaborate, please acknowledge your collaborators in the write-up for each problem. *If you obtain a solution with help from a book, paper, a website, or any other source, please acknowledge your source. You are not permitted to solicit help from other online bulletin boards, chat groups, newsgroups, or solutions from previous offerings of the course.*

## Due date

The assignment is due via Crowdmark by 8:59:59pm on September 22, 2022. Late assignments will not be accepted.

## Office hours

Office hours will take place online via the Gather.town platform; see the link on LEARN under Contents → Course Information → Office hours.

- Monday September 12 11am–12pm
- Thursday September 15 1–2pm
- Monday September 19 11am–12pm
- Tuesday September 20 11am–12pm
- Wednesday September 21 11am–12pm
- Wednesday September 21 2–3pm
- Wednesday September 21 4–5pm
- Thursday September 22 1–2pm

## Changelog

- Fri. Sep. 9: Full assignment posted.
- Sat. Sep. 10: Revised ciphertexts for question 1.
- Mon. Sep. 19: Change 4(a) so that the scheme you rely only needs to be semantically secure against a *ciphertext-only* attack.