C&O 487/687: Assignment #1 Solutions

Due date: September 26, 2024 (Thursday)

Notes and Instructions:

- Some questions may ask you to write a computer program, or you as part of solving a question you might write a computer program to help you. Include the source code of your program in the contents of the PDF / Word file / screenshot that you submit to Crowdmark. We will not run your code, but we may read it and allocate marks for it, so please include a few comments so we can understand what you've done.
- For questions on this assignment where we give you messages/ciphertexts to work with, you can assume the following:
  - All plaintexts are passages of "typical" English text, written in a similar style, without any intentional abnormalities.
  - To encode a message for encryption, all punctuation and whitespace will be removed, so that the plaintext is a string in $\{A, \ldots, Z\}^*$.
  - If the characters of the plaintext need to be represented as numbers during encryption/decryption operations, they will be represented as integers modulo 26, with the mapping $A \to 0, B \to 1, \ldots, Z \to 25$.
- Some questions use randomization to customize to you specifically. Please include your max-8-character UW user id (`b54khan`) at the beginning of your answer so we can look up your custom solution.

1. [17 marks] **Cryptanalysis of historical ciphers**

   Please include your max-8-character UW user id (`b54khan`) 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_f24/a1q1ciphertexts.zip`.

   Your folder contains 4 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 four ciphers used, in random order in your set, are:

   - shift cipher
   - substitution cipher
   - Vigenère cipher, for an unknown block length between 6 and 13
   - transposition cipher, for an unknown block length between 6 and 13

   Your task is to determine which cipher was used for each ciphertext, and what the corresponding plaintext is, using the following steps. You may write software to do so in any language of your choice, or use online tools to assist, 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) [4 marks] For each ciphertext, using either a table or a histogram, present the single character frequencies and a description of the relevant properties of the single character frequencies.

   Solution. Character frequency for file `ctxt0.txt`:

b54khan

A: 0.0200, B: 0.0720, C: 0.0580, D: 0.0140, E: 0.0220, F: 0.0520, G: 0.0340, H: 0.0150, I: 0.0360, J: 0.0410, K: 0.0450, L: 0.0370, M: 0.0650, N: 0.0350, O: 0.0210, P: 0.0410, Q: 0.0430, R: 0.0410, S: 0.0320, T: 0.0320, U: 0.0300, V: 0.0400, W: 0.0390, X: 0.0310, Y: 0.0710, Z: 0.0330,

Description of the relevant properties of single character frequencies for `ctxt0.txt`: The distribution is closer to uniform than all the others.

Character frequency for file `ctxt1.txt`:

A: 0.0120, B: 0.0390, D: 0.0200, F: 0.0980, G: 0.0140, H: 0.0180, I: 0.0460, J: 0.1220, K: 0.0230, L: 0.0260, M: 0.0740, N: 0.0710, P: 0.0090, Q: 0.0580, R: 0.0210, S: 0.0640, T: 0.0680, U: 0.0080, W: 0.0420, X: 0.0590, Y: 0.0910, Z: 0.0170,

Description of the relevant properties of single character frequencies for `ctxt1.txt`: The distribution is not uniform. It contains similar frequencies to English text, but shifted by some offset.

Character frequency for file `ctxt2.txt`:

A: 0.0810, B: 0.0140, C: 0.0130, D: 0.0560, E: 0.1430, F: 0.0160, G: 0.0230, H: 0.0710, I: 0.0690, J: 0.0010, K: 0.0100, L: 0.0510, M: 0.0130, N: 0.0690, O: 0.0640, P: 0.0140, R: 0.0450, S: 0.0720, T: 0.0800, U: 0.0250, V: 0.0140, W: 0.0310, X: 0.0010, Y: 0.0240,

Description of the relevant properties of single character frequencies for `ctxt2.txt`: The distribution is not uniform. It contains similar frequencies to English text.

Character frequency for file `ctxt3.txt`:

A: 0.0160, D: 0.0300, E: 0.0170, F: 0.0850, G: 0.1330, H: 0.0230, I: 0.0310, J: 0.0710, K: 0.0540, L: 0.0270, M: 0.0720, N: 0.0630, O: 0.0070, P: 0.0370, Q: 0.0010, S: 0.0100, T: 0.0730, U: 0.0450, V: 0.1130, W: 0.0110, X: 0.0570, Y: 0.0070, Z: 0.0170,

Description of the relevant properties of single character frequencies for `ctxt3.txt`: The distribution is not uniform. It contains similar frequencies to English text, but not associated to the right letters.

(b) [4 marks] Explain clearly how the single character frequencies are *expected* to look for the ciphertext from each of the four cipher algorithms used.

Use the experimental observations from part (a) to argue which ciphertext comes from which of the four ciphers.

Solution.

- Shift cipher: The distribution should not be uniform. It should contain similar frequencies to English text, but shifted by some offset, which corresponds to the key.
- Substitution cipher: The distribution should not be uniform. It should contain similar frequencies to English text, but not associated to the right letters.
- Transposition cipher: The distribution should not be uniform. It should contain similar frequencies to English text.
- Vigenère cipher: The distribution should be closer to uniform. If we looked at character frequencies for every nth letter, where n is the keylength, we would see frequencies similar to English text, but shifted by the key letter.

Thus, the ciphers are:

- `ctxt0.txt`: Vigenere cipher
- `ctxt1.txt`: Shift cipher
- `ctxt2.txt`: Transposition cipher
- `ctxt3.txt`: Substitution cipher

(c) [4 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.

Solution.

b54khan

b54khan

- For the shift cipher, we can cryptanalyze by using single character frequency analysis to find the shift.
- For the substitution cipher, we can cryptanalyse by using single character frequency analysis and sorting the most frequent ciphertext letters, then matching those to the most frequent English letters. We may have to make a few adjustments.
- For the transposition cipher, we can cryptanalyse by iterating through all possible block lengths, and then using a brute force search for that key length. Eventually, after exhausting all possible permutations, we will find the key and decrypt the ciphertext. We can also use bigram and trigram information to identify more likely permutations.
- For the Vigenère cipher, we can determine the block length B by using the index of coincidence technique from question 1, then we can use single character frequency analysis of the set of cipehrtext letters congruent to $i \bmod B$ to find the ith character of the key.

(d) [3 marks] Using whatever technique or tools you like, obtain the key and plaintext for the ciphertexts encrypted using the shift, substitution, and Vigenère cipher. It is okay if you only give the first 50 or so characters of the plaintext. It is okay if the first or 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.

Solution.

The keys are as follows:

- Shift cipher: 5
- Substitution cipher: MWAKGZDFXROPLJTSQUNVHYIBEC
- Vigenère cipher: OIFYGYRXU with block length 9

The first 50 characters of the plaintext are:

- Shift cipher: GECOMEINCOMEINSAIDSHEISNOTTHISMUCHBETTERTHANTHEFIL
- Substitution cipher: LEARNTSOMELITTLEMATTERSTHATAREYETUNKNOWNTOMESOTHES
- Vigenère cipher: DESAIDHETOTHEFATHERHAVEYOUNOOTHERDAUGHTERSNOSAIDHE

You can download the full plaintexts at `https://www.math.uwaterloo.ca/~dstebila/as/as.cgi/download/co487_f24/a1q1plaintexts.zip`.

(e) [2 marks] Suppose that, instead of using only one cipher, we decided to combine two ciphers with independent keys, by encrypting the message using the first cipher, and then encrypting that ciphertext using the second cipher. For each of the following explain why you could still break the combination.

i. Shift cipher followed by substitution cipher.
Solution. This is still just a substitution cipher.

ii. Substitution cipher followed by Vigenère cipher.
Solution. This is just B substitution ciphers, one for each ith position modulo B.

2. [6 marks] **Affine cipher**

The **affine cipher** is a symmetric key encryption scheme with the following properties:

- The plaintext space and ciphertext space is $\mathbb{Z}_{26}$, the set of integers modulo 26.
- The key space is the set of all pairs (a, b) where a and b are elements of $\mathbb{Z}_{26}$ such that $\gcd(a, 26) = 1$.
- The encryption function is $E_k(m) = am + b \bmod 26$, where the key is $k = (a, b)$.
- The decryption function is $D_k(c) = a^{-1}(c - b) \bmod 26$, where the key is $k = (a, b)$.

b54khan

Recall: For an element a modulo n, the inverse a^{-1} of a is an element of $\mathbb{Z}_n$ such that $a \times a^{-1} \equiv 1 \bmod n$. An inverse exists if and only if $\gcd(a, n) = 1$, in which case the inverse is unique modulo n. In practice, computing the inverse modulo n of a can be done efficiently, even if a and n are very large.

(a) [1 mark] What is the size of the keyspace of the affine cipher?

Solution. $26 \times \varphi(26) = 26 \cdot 12 = 312$, since there are 12 invertible elements modulo 26.

(b) [1 mark] Given a single plaintext-ciphertext pair (m, c), how many keys k are there such that $c = E_k(m)$?

Solution. For a fixed m and c, for every invertible a, there exists a single value of b such that $am + b \equiv c \bmod 26$.

(c) [1 mark] Suppose that you are allowed to carry out a chosen plaintext attack where you are allowed to have at most two plaintexts encrypted. Explain how you could recover the secret key.

Solution. Ask for the encryption of 0 and 1. You will then get b and $a+b$. It is easy to recover a and b from there.

(d) [2 mark] Since the affine cipher is not secure against message recovery under chosen plaintext attacks, let's try to salvage it by working in a large modulo, say a large prime $p > 2^{256}$. The new scheme becomes:

- The plaintext space and ciphertext space is $\mathbb{Z}_p$, the set of integers modulo p.
- The key space is the set of all pairs (a, b) where a and b are elements of $\mathbb{Z}_p$ such that $\gcd(a, p) = 1$.
- The encryption function is $E_k(m) = am + b \bmod p$, where the key is $k = (a, b)$.
- The decryption function is $D_k(c) = a^{-1}(c - b) \bmod p$, where the key is $k = (a, b)$.

Is the new scheme secure against an exhaustive key search attack?

Is the new scheme secure against a chosen plaintext attack? Explain why or why not.

Solution. There are now $p \cdot (p - 1)$ possible keys, which is at least 2^{512}, making an exhaustive search unfeasible. However, the same chosen plaintext attack as in the previous problem still works.

(e) [1 mark] In addition to using large numbers, what if we decided to double encrypt? The new scheme becomes:

- The plaintext space and ciphertext space is $\mathbb{Z}_p$, the set of integers modulo p.
- The keys k_1, k_2 are two random pairs $(a_1, b_1), (a_2, b_2)$ where a_1, a_2 and b_1, b_2 are random elements of $\mathbb{Z}_p$ such that $\gcd(a_1, p) = \gcd(a_2, p) = 1$.
- The encryption function is $E_{k_2}(E_{k_1}(m)) = a_2(a_1 m + b_1) + b_2 \bmod p$.
- The decryption function is $D_{k_1}(D_{k_2}(c)) = a_1^{-1}(a_2^{-1}(c - b_2) - b_1) \bmod p$.

Is this double-encrypted affine cipher secure against a chosen plaintext attack?

Solution. Observe that $a_2(a_1 m + b_1) + b_2 = a_2 a_1 m + (b_1 + b_2)$. While we cannot obtain the exact keys, we only need to obtain $a_2 a_1$ and $b_1 + b_2$ in order to encrypt and decrypt; so effectively the key is just a pair (a', b') like in the previous part of the question. The same attack as above will give us just that.

3. [6 marks] **Pseudorandom bit generators.**

Most programming languages include pseudorandom bit generators so that users can generate random numbers. For example, the C programming language has the `rand()` function, and Java has the `java.util.Random` class. Both of these functions, as well as analogous functions in many other programming languages[1] use a function called a *linear congruential generator (LCG)*, which is constructed as follows.

[1] See `https://en.wikipedia.org/wiki/Linear_congruential_generator#Parameters_in_common_use`

b54khan

- A modulus M is fixed as part of the specification of the system.
- The seed key k is a triple (a, b, X_0) where a and b are random elements of $\mathbb{Z}_M$ such that $\gcd(a, M) = 1$ and X_0 is a random element of $\mathbb{Z}_M$.
- The initialization function for the pseudorandom bit generator saves (a, b, X_0) as the initial state.
- The update and output function for the pseudorandom bit generator takes as input a state (a, b, X_i) (starting with $i = 0$), and produces an output state (a, b, X_{i+1}) where $X_{i+1} = aX_i + b \bmod M$, and outputs X_{i+1} as the next partial output of the pseudorandom generator.

Suppose we used the LCG to construct a symmetric key encryption scheme as follows. Fix modulus $M = 26$. Let the plaintext space and ciphertext space be $\{A, \ldots, Z\}^*$, in other words arbitrary-length strings of English letters. The encryption key is a randomly chosen seed key satisfying the conditions above. The encryption algorithm is as follows:

```
$E_k(m)$
for $i = 1, \ldots, |m|$ :
    $X_i \leftarrow aX_{i-1} + b \bmod 26$
    $c_i \leftarrow m_i + X_i \bmod 26$
return $c = (c_1, \ldots, c_{|m|})$
```

(a) [2 marks] Write pseudocode for the decryption algorithm.

Solution.

```
$D_k(c)$
for $i = 1, \ldots, |c|$ :
    $X_i \leftarrow aX_{i-1} + b \bmod 26$
    $m_i \leftarrow c_i - X_i \bmod 26$
return $m = (m_1, \ldots, m_{|c|})$
```

(b) [2 marks] Is this scheme secure against a chosen plaintext attack? If so, explain why. Otherwise, propose an attack.

Solution. There are at least two possible attacks. The simplest attack is to do an exhaustive key search as there are only $12 \times 26 \times 26 = 8112$ possible keys. Also, if we obtain the ciphertext for a plaintext of at least three characters, we can subtract the plaintext from the ciphertext to obtain $X_i, aX_i + b, a^2X_i + ab + b$. This is a system of 3 equations in 3 unknowns, which we can solve.

(c) [2 marks] Would the scheme be secure if we worked with a larger modulus, say M being a prime larger than 2^{256}?

Solution. We could no longer carry out an exhaustive key search, but we could still use the second approach in the previous part.

When programming any kind of security-related code, you should make sure that you are using a cryptographically secure pseudorandom number generator, rather than a weak pseudorandom number generator like the one described above. Carefully check your programming language's documentation to find out what functions are cryptographically secure pseudorandom number generators and how to use them safely. For example,

- In Java: Use `java.util.SecureRandom` instead of `java.util.Random`.
- In Python: Use the `secrets` module instead of the `random` module.

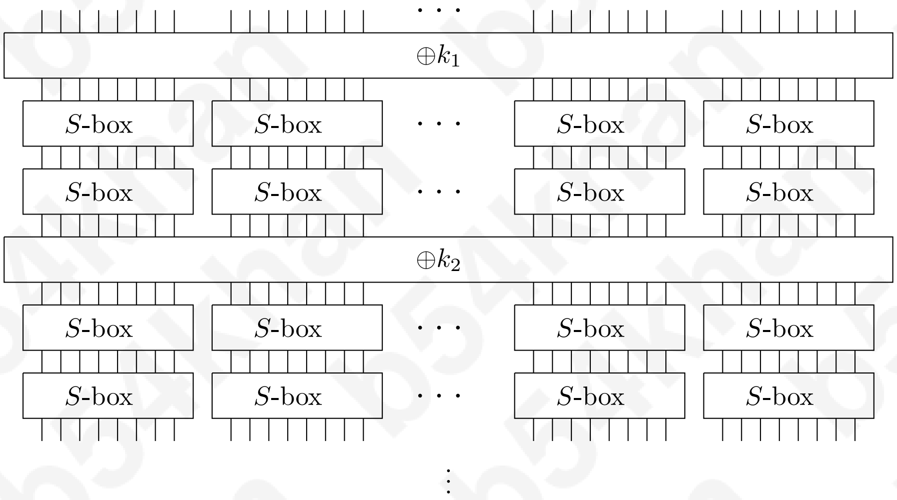
b54khan

- In C: Use an external library like OpenSSL or on Unix carefully read from `/dev/urandom`, instead of using `rand()`.

4. [8 marks] **Substitution Permutation Networks.**

As you know, there is an intense rivalry between math and engineering students at the University of Waterloo. A plot is underway to steal Math Orientation's 40-foot tall pink tie by a secretive sub-committee of the Engineering Society called the Awesome Engineering Squad (AES). In order to plan their dastardly plots, the Awesome Engineering Squad has decided to encrypt all of their communications.

However, the leader of the AES (who, it should be noted, has not taken CO 487) was recently told by a psychic that a disastrous attack on substitution permutation networks was on the horizon, since permutations are so easy to reverse. Consequently, the leaders of the AES have decided to take matters into their own hands, and create a new cipher: NVP (Not Very Permute-y). The NVP cipher works just like an substitution-permutation network, except each permutation is replaced with more S-boxes, so that the resulting cipher looks something like this:



(14 more rounds for a total of 16 rounds)

It takes as input a 128-bit message block and a 128-bit key, and outputs a 128-bit ciphertext. It expands the 128-bit key into sixteen 128-bit round keys $k_1, \ldots, k_{16}$. It has 16 rounds. In each round $i = 1, \ldots, 16$:

- the round key k_i is XORed into the state,
- then the bits are sent through the S-boxes in groups of 8,
- then the bits are sent through the S-boxes AGAIN, still in groups of 8.

Note that all the S-boxes are the same substitution.

(a) [4 marks] Describe how an adversary can totally break the NVP cipher under a chosen plaintext attack using significantly fewer operations than exhaustive key search would.

Solution. Since no permutation is used, the bits of each 8-bit piece of the message that go through each S-box are never mixed with the other 8-bit pieces of the message that go through other S-boxes. So, the message is encrypted in 8-bit blocks. The encryption of $m = (m_1, \ldots, m_{16})$ can be viewed as

$$E(k, m) = (E_1(k, m_1), \ldots, E_{16}(k, m_{16})),$$

where each $E_i(k, m) : \{0,1\}^{128} \times \{0,1\}^8 \to \{0,1\}^8$ is an invertible function. The attack is as follows:

- Create 16 lookup tables (initialized to be empty).
- For each $a \in \{0,1\}^8$, query the oracle for the encryption of $(a, a, \ldots, a)$ to receive $E(k, (a, a, \ldots, a)) = (z_1, z_2, \ldots, z_{16})$.
- For each i, store (z_i, a) in the appropriate lookup table, and sort the entries by z_i.
- To decrypt a ciphertext $(c_1, \ldots, c_{16})$, we can separately look up each c_i in the tables to determine the plaintext.

(b) [2 marks] Would adding more rounds help? Why or why not?

Solution. No. The same attack can be used (no modifications needed).

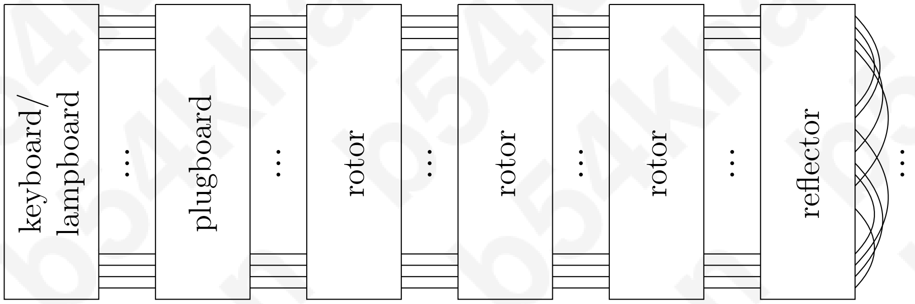
(c) [2 marks] What would be the effect of using S-boxes which are 128 bits wide, compared to 8 bits wide? Why is this not done in practice?

Solution. The attack described above would not work in this case, as it would require making 2^{128} queries to the oracle.

We don't use 128-bit wide S-boxes in practice because the scheme would be impractically inefficient: a good S-box is effectively a lookup table, so a 128-bit wide S-box would be a lookup table with 2^{128} entries, which is impractically big.

5. [9 marks] **The Engima machine and IND-CPA security.**

Perhaps the most famous cipher is the Enigma code, used by the Germans in WWII. Encryption and decryption were performed using a device called the Enigma machine. Very generally speaking, the Enigma machine is arranged as follows, where each line represents an electrical wire:



When the Enigma machine is configured, it forms a circuit leading from the keys on the keyboard to the lamps on the lampboard. When an Axis member wishes to encrypt a letter, they type that key, which sends current flowing through the plugboard, the rotors, the reflector, back through the rotors, back through the plugboard, and finally to the lampboard, where the lamp corresponding to the encryption of that letter will light up. For this symmetric key cryptosystem, the secret key is the choice of rotors (there are many different rotor options, but Enigma machines only use three at a time), and the wiring of the rotors, plugboard, and reflector. Randomness is introduced by the operator (who is encrypting a message) by choosing the initial position of the rotors. This should be different for every message, and the settings are sent in the clear before the encrypted ciphertext is sent. For a more detailed description of each component:

- **Keyboard:** This consists of the 26 letters of the alphabet, and is the part of the machine which takes input. Axis members encrypt messages letter-by-letter, by pressing the key corresponding to each letter they wish to encrypt.
- **Plugboard:** This swaps certain letters, in a way determined by the secret key. In most configurations, 10 pairs of letters were chosen (the letters in each pair were swapped with each other), and the remaining 6 letters were left untouched.

b54khan

b54khan

- **Rotors:** Each individual rotor works as a substitution cipher. However, they have the ability to rotate, and (some) rotors do so each time a new key is pressed. Together, they form a *polyalphabetic substitution cipher*[2] which is very difficult (not impossible, but very time-consuming) to crack by brute force. It is important to note that, due to the multiple substitution alphabets, it might be the case that multiple instances of the same letter in a word are mapped to different letters by the encryption function, and that two different letters in the word are mapped to the same letter by the encryption function.
- **Reflector:** This pairs up the 26 letters of the alphabet, and connects each pair with a wire to send the current back through the rotors and plugboard along a different path than the one it came from. The main purpose of the reflector is to give Enigma the property that encryption is its own inverse — that is, decryption can be performed by running the encryption algorithm on a ciphertext (under the same settings that it was encrypted under). This was a very desirable property for convenience and practicality reasons.
- **Lampboard:** This consists of 26 lamps, each corresponding to a letter of the alphabet. The circuit formed by the plugboard, rotors, and reflector connects every key on the keyboard to exactly one lamp on the lampboard (namely, the one which corresponds to the encryption of the key pressed), and this lamp lights up when its corresponding key is pressed.

(a) [6 marks] Which of the following are possible encryptions of the message "STEBILA"? That is, for which of the following could there exist a secret key such that the encryption of "STEBILA" on an Enigma machine configured in the way specified by the key, would produce the ciphertext listed? Give a brief explanation in each case (you can group answers with the same explanation together).

(i) ESTUPINAN
(ii) MOKRANI
(iii) RIVERA
(iv) REYNES
(v) STECKEL
(vi) SWANSON
(vii) GOOOOSE

Hint: There should be three categories: POSSIBLE, IMPOSSIBLE because reason X, IMPOSSIBLE because reason Y, and reasons X and Y should have very different flavours.

Solution.
Notice two things from the Enigma machine specification:

- Messages are encrypted letter-by-letter, so the length of the ciphertext will be the same as the length of the message.
- The reflector guarantees that a letter cannot be encrypted to itself (i.e., if position i in the message is "A" then position i in the ciphertext cannot be "A").

However, all other encryptions are possible (in theory). So, "STEBILA" could encrypt to "MOKRANI" and "GOOOOSE," but not "ESTUPINAN," "RIVERA," or ‘REYNES" (they're not the right length), and also not "STECKEL" or "SWANSON" ("S" encrypts to itself in both cases).

(b) [3 marks] Show that Enigma does not satisfy IND-CPA security. To do this, you must write an adversary which can win the IND-CPA security experiment on this cryptosystem with probability noticeably greater than $\frac{1}{2}$ (and justify why this is the case). You may assume that every possible encryption of a letter occurs with equal probability.

Solution. Our adversary $\mathcal{A}$ acts as follows:

[2] The specifics of what a polyalphabetic cipher is are not important for this question, but essentially they are substitution ciphers that use multiple substitution alphabets. The Vigenère cipher is a simplified, special case of a polyalphabetic cipher.

- The adversary picks $m_0 = A$ and $m_1 = B$, and sends these values to the challenger.
- When the adversary receives back the challenge ciphertext c, they check if $c = B$. If so, they send their guess $b = 0$ to the challenger. Otherwise, they send their guess $b = 1$ to the challenger.

First, suppose that $b = 0$. Then $\mathcal{A}$ only wins if $c = B$. There is a $\frac{1}{25}$ probability of encrypting $m_0 = B$ So,

$$\Pr(\mathcal{A} \text{ wins} \mid b = 0) = \frac{1}{25}.$$

Now, suppose that $b = 1$. Then, since Enigma cannot encrypt a letter to itself (and $m_1 = B$), $\mathcal{A}$ always wins in this case. So,

$$\Pr(\mathcal{A} \text{ wins} \mid b = 1) = 1.$$

Putting this together gives us that

$$\begin{aligned}
\Pr(\mathcal{A} \text{ wins}) &= \Pr(\mathcal{A} \text{ wins} \mid b = 0) \Pr(b = 0) + \Pr(\mathcal{A} \text{ wins} \mid b = 1) \Pr(b = 1) \\
&= \frac{1}{2} \Pr(\mathcal{A} \text{ wins} \mid b = 0) + \frac{1}{2} \Pr(\mathcal{A} \text{ wins} \mid b = 1) \\
&= \frac{1}{2} \cdot \frac{1}{25} + \frac{1}{2} \cdot 1 \\
&= \frac{1}{2} + \frac{1}{50}
\end{aligned}$$

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 11:59:59pm on September 26, 2024. Late assignments will not be accepted.

b54khan