Problem Set 1

CNT 5412 — Dr. Sudhir Aggarwal — Fall 2018

David Miller

Problem 1

You are given the following ciphertext:

```
GKYBEEYDYQVONNRBSNFPLBVRBCOGRSBNKIOEQCYVYEYBNYQUVILZBREBSQFEBJYQISKIONYBVVYNGKYVBGGIVSYPN
BVDOYQRSBJIOVGURERSDUVRQBPGKBGNYYTNGIFBRSGKYVBNBKOLCEYNGOQYSGAKINYRSGYVBJGRISNARGKBLYVRJB
SNBSQVONNRBSNBVYIWYVCEIASCPFVINYJOGIVNAKIRSQRJGYQKYVUIVBJGRSDREEYDBEEPBNBUIVYRDSBDYSGNFYJ
RURJBEEPCOGRSBQYUYSNYGYBLQYNJVRCYNKYVVYEBGRISNKRFGIVONNRBSCREERISBRVYBEYHBSQYVGIVNKRSAKIB
EEYDYQEPLYSGIVYQKYVRSBGGYLFGNGIRSUREGVBGYVYFOCERJBSFIERGRJNBNBJGOBEEPZONGBUVRYSQKYVEBAPYV
NBENIRSJEOQYQBGYHGLYNNBDYJKBRSRSAKRJKBLBVVRYQDOSVRDKGNBQWIJBGYAKILCOGRSBBEEYDYQEPIUUYVYQG
INEYYFARGKRSBSBGGYLFGGIDBRSFIERGRJBERSUEOYSJYNBRQKYAIOEQSYWYVNEYYFARGKBVYQKYBQERTYCOGRSBC
OGRSBKBNCYYSRSZBREURVNGRSABNKRSDGISSIARSBEYHBSQVRBWRVDRSRBNRSJYZOEPURUGYYSAKYSNKYFEYBQYQS
IGDOREGPGIGKYUIVYRDSBDYSGJKBVDYNRSQJUYQYVBEJIOVGCOGRSBNKIONYBVVYNGVYMOYNGAREECYQRNJONNYQB
GBJIOVGKYBVRSDISNYFGYSGKYZOQDYNBRQGKYNRCYVRBSCIVSUIVLYVBLYVRJBSOSRWYVNRGPDVBQOBGYNGOQYSGB
EEYDYQEPCOREGVYFOCERJBSFBVGPJISSYJGRISNGKVIODKDOSVRDKGNDVIOFNRSJEOQRSDGKYSVBBSQGKVIODKGKY
SBGRISBEFVBPYVCVYBTUBNGBNBABPGIFONKFVIVONNRBSRSGYVYNGNFVINYJOGIVNJKBVDY
```

(a) First determine the relative frequency of the letters and compare this to the frequency of normal English. Does this seem to be a substitution cipher that maintains frequencies? Show your work. Now assume that you learn that this is a monoalphabetic cipher. What ciphertext letter do you think correspond to "e" in the plaintext? Examining the ciphertext and considering bigrams and trigrams, try to determine what ciphertext letters correspond to "t" and "h" in the plaintext. Your main problem is to determine the plaintext by using the statistical frequency of letters approach. Show your work.

Running a simple Python script, we parse the ciphertext and obtain a frequency count of the letters. We can compare this to the the frequency of the letters of the English alphabet.

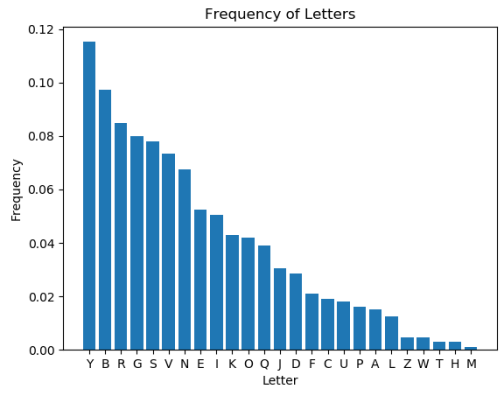

Figure 1: Frequencies of ciphertext in Problem 1

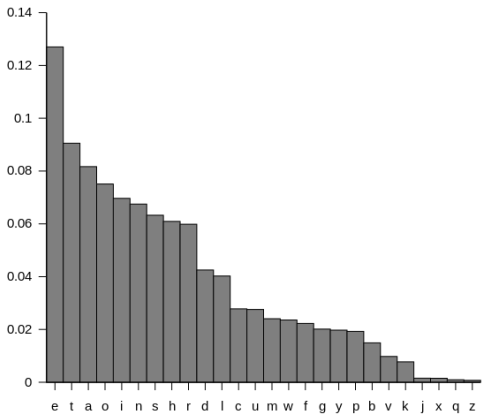

Figure 2: Frequencies for English alphabet

After plotting the same for digrams and trigrams, the plaintext was able to uncovered. I used the following heuristics:

- If the frequency of single letter, digram, and trigram matched for the same letter, I mapped it accordingly.
- If the frequency of single letter and either digram or trigram matched for the same letter, I mapped it accordingly.

- If digram and trigram mathced for the same letter, but not single letter, I mapped it accoridngly. If this did not work I brute forced the remaining possibilities (small space of possibilities at this point).

After doing the above I was able to come to the following mapping of cipher to English letters.

English	A	B	C	D	E	F	G	H	I	J	K	L	M
Cipher	B	C	J	Q	Y	U	D	K	R	Z	T	E	L
English	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
Cipher	S	I	F	M	V	N	G	O	W	A	H	P	X

Figure 3: English letters and their corresponding cipher letters.

(b) Assume now that the monoalphabetic cipher you have determined was generated using transposition as follows: first choose a keyword, (for example "CIPHER") and write the keyword as the first row. Then write the remaining letters in the following rows. The monoalphabetic cipher to use is then read-off by the columns left to right. Using the keyword "CIPHER" we get:

```
C  I  P  H  E  R
A  B  D  F  G  J
K  L  M  N  O  Q
S  T  U  V  W  X
Y  Z
```

which yields the cipher:

```
plain:  a b c d e f g h i j k l m n o p q r s t u v w x y z
cipher:  C A K S Y I B L T Z P D M U H F N V E G O W R J Q X.
```

Determine the keyword. Show your work.

We want the first letter of the keyword to be B since A $\rightarrow$ B. We also want the letter A somewhere in the keyword so that B $\not\rightarrow$ A. Assuming a length of 6, we want the letter I also in the keyword so that C $\rightarrow$ J.

Problem 2

Consider the pseudo-code [and diagram] of an encryption algorithm using a Feistel-type cipher:

Algorithm 1 Encryption

Input: plaintext = $(L[0], R[0])$
Output: ciphertext = $(L[16], R[16])$

1: **for** $i = 1$ to 16 **do**
2: $\quad L[i] \leftarrow R[i-1]$
3: $\quad R[i] \leftarrow L[i-1] \oplus F(K[i-1],\, R[i-1])$
4: **end for**

Figure 4: Encryption algorithm.

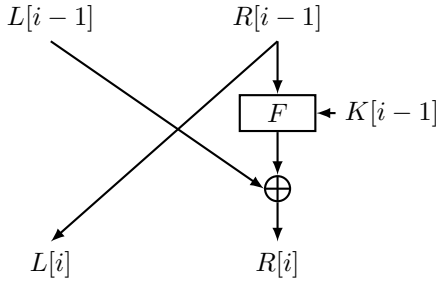

Figure 5: One round of encryption.

Describe the pseudo-code of the corresponding decryption algorithm for this cipher.

Solution: If we look at figure 1 (below), we can visualize a round of this Feistel network. The easiest thing to notice is that $L[i] = R[i-1]$, that is the left half in the next round is the right half from the previous round. We can use this fact to get $R[i-1]$ trivially and use it to compute $F(K[i-1], R[i-1])$. Now we know from encryption that $R[i] = L[i-1] \oplus F(K[i-1], R[i-1])$ and we also know that $(\{0,1\}^n, \oplus)$ forms an abelian group and therefore each element must have an inverse. From this we can easily compute $L[i-1]$ by XORing by $F(K[i-1], R[i-1])$ to get $L[i-1] = R[i] \oplus F(K[i-1], R[i-1])$. We do this for 16 rounds. Figures 6 and 7 below give a one-round decryption.

Algorithm 2 Decryption

Input: ciphertext = $(L[16], R[16])$
Output: plaintext = $(L[0], R[0])$

1: **for** $i = 16$ to 1 **do**
2: $\quad R[i-1] \leftarrow L[i]$
3: $\quad L[i-1] \leftarrow R[i] \oplus F(K[i-1],\, R[i-1])$
4: **end for**

Figure 6: Decryption algorithm.

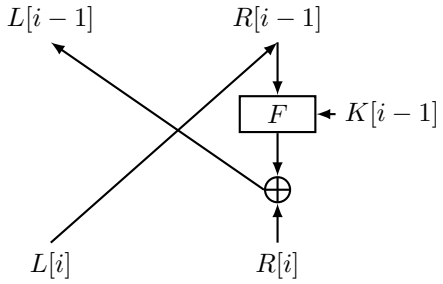

Figure 7: One round of decryption.

Problem 3

Consider a 64 bit Block Encryption algorithm E. Suppose you have encrypted a 64 byte message $M_1, M_2, \ldots M_{64}$. Show / explain using diagrammatic forms how you would decrypt the message using CBC and OFB-16. Use an 8 byte IV: $I_1, I_2, ...I_8$.

Solution: Consider the pseudo-code and diagram of CBC encryption:

Algorithm 3 CBC-Encryption

Input: message $= M_1, M_2, \ldots, M_{64}$,
$IV = I_1, I_2, \ldots, I_8$
Output: ciphertext $= C_1, C_2, \ldots, C_8$

1: $C_0 \leftarrow IV$
2: **for** $i = 1$ to m **do**
3: $\quad C_i \leftarrow E_K(M_{8i-7:8i} \oplus C_{i-1})$
4: **end for**

Figure 8: CBC-Encryption algorithm.

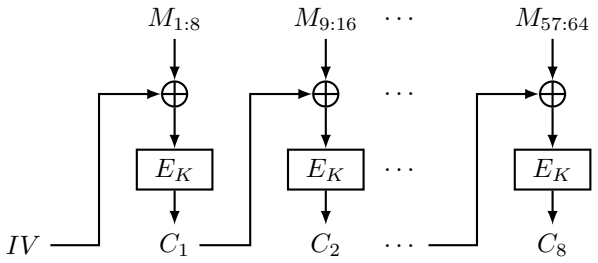

Figure 9: CBC-Encryption diagram.

We must reverse the computation in line 3 for Algorithm 3 to retrieve the message. The key to the is knowing that $E_K(\cdot)$ is a block cipher and therefore has an inverse function $E_K^{-1}(\cdot)$. Applying the inverse to both sides of the equation in line 3 we get $E_K^{-1}(C_i) = E_K^{-1}(E_K(M_{8i-7:8i} \oplus C_{i-1})) = M_{8i-7:8i} \oplus C_{i-1}$. Using the fact that $(\{0,1\}^n, \oplus)$ forms an abelian group we can XOR both sides by the inverse of C_{i-1} to get $M_{8i-7:8i} = E_K^{-1}(C_i) \oplus C_{i-1}$. Given below are the encryption/decryption diagrams for CBC and the pseudo code for the decryption.

Algorithm 4 CBC-Decryption

Input: ciphertext $= C_1, C_2, \ldots, C_8$,
$IV = I_1, I_2, \ldots, I_8$
Output: message $= M_1, M_2, \ldots, M_{64}$

1: **for** $i = 1$ to m **do**
2: $\quad M_{8i-7:8i} \leftarrow E_K^{-1}(C_i) \oplus C_{i-1}$
3: **end for**

Figure 10: CBC-Decryption algorithm.

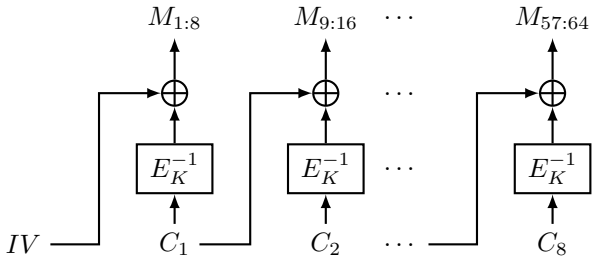

Figure 11: CBC-Decryption diagram.

OFB encryption is defined as

$$C_i = P_i \oplus I_i, \quad I_i = E_K(I_{i-1}), \quad I_0 = E_K(IV).$$

Using the inverse property of a block cipher, $E_K^{-1}(E_K(P)) = P$, and the identity element property for XOR, $x \oplus x = 0$, we can easily derive the decryption algorithm:

$$P_i = C_i \oplus I_i, \quad I_i = E_K(I_{i-1}), \quad I_0 = E_K(IV).$$

Diagram and pseudo code for OFB omitted for brevity.

Problem 4

If a bit error occurs in the transmission of a ciphertext character in 8-bit CFB mode, how far does the error propagate? Explain.

Solution: Before starting the problem, let us take a look at the CFB encryption scheme.

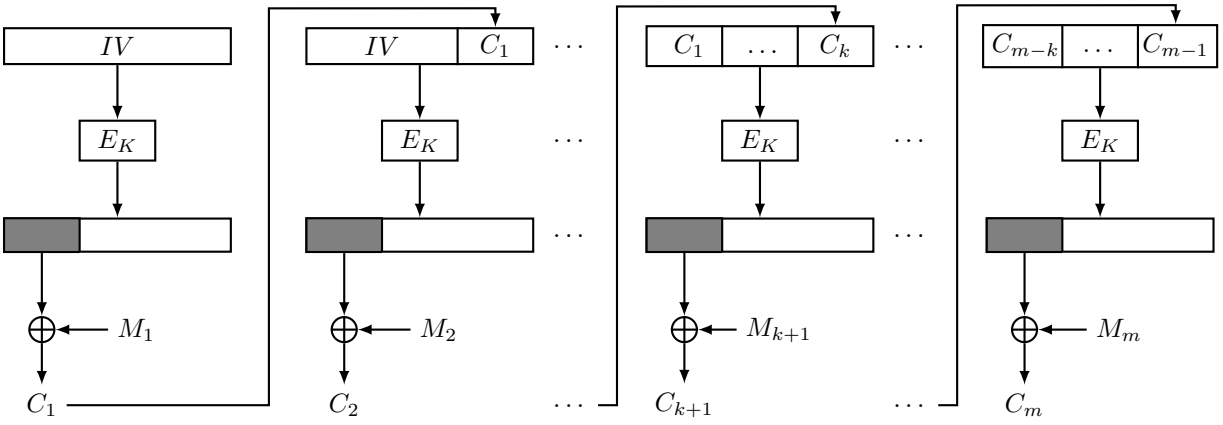

Figure 12: CFB encryption.

We can see from the encryption above that if an error occurs at some ciphertext C_i then it propagates forward since it is used in the shift register fed into E_K. Assuming ciphertext C_i incurs an error, it is easy to see that it stays in the shift register for $n/8$ rounds, where n is the block size used for the shift register. If the block size is 64 bit, then the current round and the next 8 rounds will have the error. If the block size is 128 then the current round and the next 16 rounds will have the error. Therefore there will $n/8 + 1$ rounds affected by the error, including the current round.

Problem 5

Kerberos used a modified version of CBC called the propagating cipher block chaining mode (PCBC) designed to cause small changes to propagate indefinitely when decrypting (and encrypting). The encryption algorithm is described as follows:

$$C_i = E_K[P_i \oplus P_{i-1} \oplus C_{i-1}], \quad P_0 \oplus C_0 = IV$$

What is the corresponding decryption algorithm? Explain why error in one block would propagate indefinitely. However, what happens if the intruder swaps two blocks? Explain with an example.

Solution Using the inverse property of a block cipher, $E_K^{-1}(E_K(P)) = P$, and the identity element property for XOR, $x \oplus x = 0$, we can easily derive the decryption algorithm:

$$P_i = E_K^{-1}(C_i) \oplus P_{i-1} \oplus C_{i-1}, \quad P_0 = C_0 \oplus IV.$$

Assume some ciphertext C_k incurred an error ϵ and is now $C'_k = C_k \oplus \epsilon$. We then have the following decryption

$$\begin{aligned}
P_0 &= C_0 \oplus IV \\
P_1 &= E_K^{-1}(C_1) \oplus C_0 \oplus P_0 \\
&\vdots \\
P'_k &= E_K^{-1}(C'_k) \oplus C_{k-1} \oplus P_{k-1} \\
P'_{k+1} &= E_K^{-1}(C_{k+1}) \oplus C'_k \oplus P'_k \\
&\vdots \\
P'_n &= E_K^{C_n} \oplus C_{n-1} \oplus P'_{n-1}
\end{aligned}$$

where n is the total number of decryption rounds. As we can see the error in C'_k brings about an error in P_k since $E_K^{-1}(C'_k) \neq E_K^{-1}(C_k)$. The error propagates indefinitely because each plaintext P_i relies on the previous plaintext P_{i-1}.

Assume blocks C_k and C_{k+1} were swapped, then we would have plaintexts $P'_k = E_K^{-1}(C_{k+1}) \oplus C_{k-1} \oplus P_{k-1}$, $P'_{k+1} = E_K^{-1}(C_k) \oplus C_{k+1} \oplus P_k$, and $P'_{k+2} = E_K^{-1}(C_{k+2}) \oplus C_k \oplus P_{k+1}$. Plaintext P'_{k+1} can be rewritten as $E_K^{-1}(C_k) \oplus C_{k+1} \oplus E_K^{-1}(C_{k+1}) \oplus C_{k-1} \oplus P_{k-1} = E_K^{-1}(C_{k+1}) \oplus C_{k+1} \oplus P_k$. Therefore we can rewrite plaintext P'_{k+2} as $E_K^{-1}(C_{k+2}) \oplus C_k \oplus (E_K^{-1}(C_{k+1}) \oplus C_{k+1} \oplus P_k) = E_K^{-1}(C_{k+2}) \oplus C_{k+1} \oplus P_{k+1} = P_{k+2}$. Therefore, changing adjacent blocks only affects the two corresponding decryptions but nothing after it.

Problem 6

The pseudo-random stream of blocks generated by a 64-bit Output Feedback Mode cipher must eventually repeat since there are only 2^{64} different blocks that can be generated. Does a sequence necessarily repeat once a block repeats? Can we determine which block is the first block to repeat? Is it IV or $K\{IV\}$ or some other block? Explain your answer.

Solution IV will be the first block to repeat. Let B_i denote the i-th encryption of IV where B_{i+1} is the encryption of B_i and B_i the decryption of B_{i+1}. Now let $B_k = B_j$ be the first block to repeat for $j < k$. When $j = 1$ the proof is done trivially. When $j > 1$ we have that $B_{k-1} = B_{j-1}$ since $B_k = B_j$. This is a contradiction since B_k is not the first repeated block. Therefore $B_k = B_1 = IV$.

Problem 7

Consider a multiplicative cipher defined by $C = (P \times k) \mod 26$ analogous to the generalized (additive) Caesar cipher discussed in class. What is the encryption of "hello" using the key $k = 5$? What is the formula for decryption and why does it work? Would $k = 2$ be a good key; why or why not?

Solution: Let $\Sigma = \{\text{a}, \text{b}, \ldots, \text{z}\}$. The multiplicative cipher $C : \Sigma \to \Sigma$ maps "hello" using the following computations:

$$C(\text{h}) = (7 \times 5) \mod 26 = 9, \quad C(\text{e}) = (4 \times 5) \mod 26 = 20$$
$$C(\text{l}) = (11 \times 5) \mod 26 = 3, \quad C(\text{o}) = (14 \times 5) \mod 26 = 18.$$

Substituting $i - 1$ for the i-th letter in Σ we get that "hello" = "judds" under C.

Since C is defined over $(\mathbb{Z}_{26}, \times)$, the decryption C^{-1} leverages the multiplicative inverse k^{-1} to extract P. That is $C \times k^{-1} = P \times k \times k^{-1} = P$. Therefore we may use an algorithm such as the Extended Euclidean Algorithm to compute k^{-1} and use this to compute P given $C(P)$. Since 5 and 26 are coprime the key k generates Σ.

A key value of $k = 2$ would not be a good choice since $\gcd(2, 26) = 2$. This means that the key k generates only the even letters of Σ and decryption is no longer possible since C is not one-to-one.

Problem 8

Fill in the remainder of the table.

Solution:

MODE	**ENCRYPT**	**DECRYPT**
ECB	$C_j = E_K[P_j], \quad j = 1, \ldots, N$	$P_j = D_K[C_j], \quad j = 1, \ldots, N$
CBC	$C_1 = E_K[P_1 \oplus IV]$ $C_j = E_K[P_j \oplus Cj - 1], \quad j = 2, \ldots, N$	$P_1 = D_K[C_1] \oplus IV$ $P_j = D_K[C_i] \oplus C_{i-1}, \quad j = 2, \ldots, N$
CFB	$C_1 = E_K[IV] \oplus P_1$ $C_j = E_K[C_{j-1}] \oplus P_j, \quad j = 2, \ldots, N$	$P_1 = E_K[IV] \oplus C_1$ $P_j = E_K[C_{j-1}] \oplus C_j, \quad j = 2, \ldots .N$
OFB	$C_0 = P_0 \oplus E_K[IV]$ $C_j = P_j \oplus I_j, \quad j = 1, \ldots, N$ $I_j = E_k[I_{j-1}], \quad j = 1, \ldots, N$ $I_0 = E_K[IV]$	$P_0 = C_0 \oplus E_K[IV]$ $P_j = C_j \oplus I_j, \quad j = 1, \ldots, N$ $I_j = E_k[I_{j-1}], \quad j = 1, \ldots, N$ $I_0 = E_K[IV]$
CTR	$C_0 = E_K[IV] \oplus P_0$ $C_j = E_K[IV + j] \oplus P_j, \quad j = 1, \ldots, N$	$P_0 = C_0 \oplus E_K[IV]$ $P_j = E_K[IV + j] \oplus C_j, \quad j = 1, \ldots, N$