Arnel Jerome Adviento - 10130641
Introduction to Cryptography CPSC 418 Fall 2016
Department of Computer Science
University of Calgary

October 7, 2016

HOME WORK #[NUMBER]

Problem	Marks
1	
2	
3	
4	
5	
6	
7	
Total	

Problem 1. (A substitution cipher cryptanalysis, 10 marks plus 1 bonus mark)

(a) Letter Frequency:

A: 45	B: 2	C: 46	D: 19	E: 26	F: 30	G: 13
H: 4	I: 55	J: 71	K: 28	L: 18	M: 7	N: 30
O: 84	P: 56	Q: 16	R: 58	S: 15	T: 56	U: 56
V: 8	W: 1	X: 135	Y: 2	Z: 23		

(b) Plain Text:

Case was twenty-four. At twenty-two, he'd been a cowboy, a rustler, one of the best in the Sprawl. He'd been trained by the best, by McCoy Pauley and Bobby Quine, legends in the biz. He'd operated on an almost permanent adrenaline high, a byproduct of youth and proficiency, jacked into a custom cyberspace deck hat projected his disembodied consciousness into the consensual hallucination that was the matrix. A thief, he'd worked for other, wealthier thieves, employers who provided the exotic software required to penetrate the bright walls of corporate systems, opening windows into rich fields of data.

He's made the classic mistake, the one he's sworn he'd never make. He stole from his employers. He kept something for himself and tried to move it through a fence in Amsterdam. He still wasn't sure how he'd been discovered, not that it mattered now. He'd expected to die, then but they only smiled. Of course he was welcome, they told him, welcome to the money. And he was going to need it. Because–still smiling–they were going to make sure he never worked again.

They damaged his nervous system with a wartime Russian mycotoxin.

(c) William Gibson

$\longrightarrow$ Answer

Problem 2. (Superencipherment for substitution ciphers, 12 marks)

(a) (i) Let $E_k(M) = M + K(mod\ 26)$ be the shift cipher and M represents the plain text and K the key to shift each letters by.
Let there be keys K_1 and K_2.

We use K_1 to encrypt M in a shift cipher $E_{k1}(M) = M + K_{k1}(mod\ 26)$
Then we apply another shift cipher of key K_2 on $E_{k1}(M)$ resulting in $E_{k2}(E_{k1}(M)) = (M + K_1(mod\ 26)) + K_2(mod\ 26)$
We can simplify it to $M + ((K_1 + K_2)(mod\ 26))$
Where we let $(K_1 + K_2)$ be another key K_3.
Therefore we proved that doing the shift cipher twice just results in another shift cipher with a key of $(K_1 + K_2) = K_3$.

(ii) Let $\boldsymbol{E_k(M) = M + K(mod\ 26)}$ be the shift cipher and $\boldsymbol{M}$ represents the plain text and $\boldsymbol{K}$ represent the key to shift each letters by.
base case (K = 0):

$$\boldsymbol{E_{k0}(M) = M + K_0(mod\ 26)}$$
$$\boldsymbol{E_{k0}(M) = M + 0}$$

base case (K = 1):

$$\boldsymbol{E_{k1}(M) = M + K_1(mod\ 26)}$$
$$\boldsymbol{E_{k1}(M) = M + 1}$$

Assume there is a K, greater than or equal to 1, such that
$\boldsymbol{E_{k1}(M) = M + (K_1)(mod\ 26)}$.
Now we want to prove that $\boldsymbol{E_{k1+1}(M) = M + (K_1 + 1)(mod\ 26)}$ also holds true.
Inductive Step:

$$\boldsymbol{E_{k1+1}(M) = M + (K_1 + 1)(mod\ 26)}$$
$$\boldsymbol{E_{k1+1}(M) = M + (K_1(mod\ 26)) + (1(mod\ 26))}$$
$$\boldsymbol{E_{k1+1}(M) = (M + K_1(mod\ 26)) + (1(mod\ 26))}$$
$$\boldsymbol{E_{k1+1}(M) = E_{k1}(M) + (1(mod\ 26))}$$
$$\boldsymbol{E_{k1+1}(M) = E_{k1}(M) + 1}$$

Therefore by induction the key of multiple encipherment with shift cipher results in another shift cipher where the new key is just the sum of all the keys used previously.

(b) (i) yes, the new keyword of using the keyword ***WORD*** and ***CIPHER***
is ***YWGKAFTLLVVU***
This key is created by adding each corresponding letter of the two keywords based on their index and repeating each keyword if necessary until both keywords are an equal length.

WORDWORDWORD
CIPHERCIPHER

YWGKAFTLLVVU

(ii) Yes, to obtain the corresponding key word you add each letter of $\boldsymbol{m_1}$ to each corresponding character to $\boldsymbol{m_2}$ based on its index in the alphabet $\boldsymbol{(0...25)}$ and modulo it by 26.
eg: $\boldsymbol{m_1.charAt(0) = T}$ and $\boldsymbol{m_2.charAt(0) = I}$. Since $\boldsymbol{I}$ is at index $\boldsymbol{9}$ we shift that many letters from $\boldsymbol{T}$ and since we modulo it by 26 the combined vigenere cipher of characters $\boldsymbol{T}$ and $\boldsymbol{I}$ is $\boldsymbol{B}$. As for the length of the new key word, it will be the least common multiple length that $\boldsymbol{m_1}$ and $\boldsymbol{m_2}$ share

⟶ *Answer*

Problem 3. (Equiprobability maximizes entropy for two outcomes, 12 marks)

(a) Let $\boldsymbol{p(X_1) = \frac{1}{4}}$ and let $\boldsymbol{p(X_2) = \frac{3}{4}}$
$\boldsymbol{H(X) = -p\ log_2(p) - (1-p)\ log_2(1-p)}$
$\boldsymbol{H(X) = -\frac{1}{4}\ log_2(\frac{1}{4}) - \frac{3}{4}\ log_2(\frac{3}{4})}$
$\boldsymbol{H(X) = 0.5 + 0.311278...}$
$\boldsymbol{H(X) = 0.811278..}$

(b) To prove that $\boldsymbol{H(X)}$ is maximal, when both outcomes are equally likely. we take the derivative of $\boldsymbol{H(X) = -p\log_2(p) - (1-p)\log_2(1-p)}$, where $\boldsymbol{p}$ is the probability outcome such that $\boldsymbol{p + (1-p) = 1}$.

$$\begin{aligned}
\boldsymbol{H(X)} &= \boldsymbol{-p\log_2(p) - (1-p)\log_2(1-p)} \\
\boldsymbol{H'(X)} &= \boldsymbol{[-\log_2(p) + p\frac{1}{p\ln(2)}] - [(-1)log_2(1-p) + (1-p)\frac{1}{(1-p)\ln(2)}]} \\
\boldsymbol{H'(X)} &= \boldsymbol{-\log_2(p) - \frac{1}{\ln(2)} + log_2(1-p) + \frac{1}{\ln(2)}} \\
\boldsymbol{H'(X)} &= \boldsymbol{-\log_2(p) + log_2(1-p)} \\
\boldsymbol{H'(X)} &= \boldsymbol{log_2(1-p) - \log_2(p)}
\end{aligned}$$

Now we set H'(X) = 0 and solve for $\boldsymbol{p}$ to find the maximum value for H(X)

$$\begin{aligned}
\boldsymbol{H'(X)} &= \boldsymbol{log_2(1-p) - \log_2(p)} \\
\boldsymbol{0} &= \boldsymbol{log_2(1-p) - \log_2(p)} \\
\boldsymbol{0} &= \boldsymbol{(1-p) - p} \\
\boldsymbol{p} &= \boldsymbol{1-p} \\
\boldsymbol{2p} &= \boldsymbol{1} \\
\boldsymbol{p} &= \boldsymbol{\frac{1}{2}}
\end{aligned}$$

(c) The maximal value of $\boldsymbol{H(X)}$ is 1, when $\boldsymbol{p(X_1) = \frac{1}{2}}$ and $\boldsymbol{p(X_2) = \frac{1}{2}}$

⟶ $\mathcal{A}$nswer

Problem 4. (Key size versus password size, 21 marks)

(a) There are 128 possible characters in the ASCII encoding, and we have a space of 8 characters, therefore that gives us $\mathbf{128^8 = 7.205759 \times 10^{16}}$

(b) (i) Since there are, **94** allowed ASCII characters and the ength of the password is exactly **8**. That means at each character slot there are **94** differently possibilities.
Therefore we get the equation:
$\mathbf{94^8} = \mathbf{6095689385410816}$
total permissible passwords = **6.1** x $\mathbf{10^{15}}$

(ii) $\frac{\mathbf{94^8}}{\mathbf{128^8}} = \mathbf{0.084594...}$ roughly about 8.5% of all ASCII are usable for passwords

(c) The entropy of a 8 character length password with 94 usable characters is:

$$\boldsymbol{H(X) = \log_2(94^8)}$$
$$\boldsymbol{H(X) = 2.436710}$$

(d) The entropy of a 8 character length password with 26 usable characters is:

$$\boldsymbol{H(X) = \log_2(26^8)}$$
$$\boldsymbol{H(X) = 37.603517}$$

(e) (i)

$$\begin{aligned} \boldsymbol{94^x} &= \boldsymbol{2^{128}} \\ \boldsymbol{\ln(94^x)} &= \boldsymbol{ln(2^{128})} \\ \boldsymbol{x\, ln(94)} &= \boldsymbol{128\, ln(2)} \\ \boldsymbol{x} &= \boldsymbol{\frac{128\, ln(2)}{ln(94)}} \\ \boldsymbol{x} &= \boldsymbol{19.5283034} \\ \boldsymbol{x} &\approx \boldsymbol{20} \end{aligned}$$

We need about a 20 characted length password to get an entropy of 128 given 94 usable characters

(ii)

$$\begin{aligned} \boldsymbol{26^x} &= \boldsymbol{2^{128}\, \ln(26^x)} & &= \boldsymbol{ln(2^{128})} \\ \boldsymbol{x\, ln(26)} &= \boldsymbol{128\, ln(2)} \\ \boldsymbol{x} &= \boldsymbol{\frac{128\, ln(2)}{ln(26)}} \\ \boldsymbol{x} &= \boldsymbol{27.2314948} \\ \boldsymbol{x} &\approx \boldsymbol{28} \end{aligned}$$

We need about a 28 characted length password to get an entropy of 128 given 26 usable characters

⟶ 𝒜nswer

Problem 7. (BONUS: Vigenere cipher cryptanalysis, 10 bonus marks)

The first step is solving this problem is to apply frequencly analysis. But since it is a Vigenere cipher which is a polyalphabetic cipher, we need to apply letter frequency analysis at each of the character depending on length of the key, given in this case which is a length of 6. Therefore in my code I created a method to read the encrypted text and output the frequency at each index of the key. For example if wanted to know the letter frequency at the first character of the key, i would only print out the letters from the crypted text such that the index of the letter in the crypted text modded by 6 is equal to 0. I therefore repeated this step a total of 6 times, displaying the letter frequency for each case in my program. After checking the frequency I created an algorithm to decrypt the code given a key by subtracting the value of the crypted letter to its corresponding letter in the key, if the value is less than 0 i would subtract starting from Z and if its greater than 0 i would add starting at A. Once i had this algorith I initially set the key to "AAAAAA" which would keep the enciphered text as is and i would replace each appropriate "A" to the appropriate key based on the letter frequency. For example for every 6th characted in the encyphered text if the most frequent letter was "Q" i would assume its an "E" in plain text then on the Vigenere tableu I would look for "E" on the left hand column and search across is until i find "Q" and find the corresponding key letter in the top column which is "Q". This is not always true as my first pass did not yield me the appropriate key. Although the key was not initially correct I was able to see portions of fimiliar words in the semi-decrypted text and I was able to find the appropriate letters which led to the appropriate key "LAUNCH"

⟶ 𝒜nswer

Submitted by Arnel Jerome Adviento - 10130641 on October 7, 2016.