Hardening Weak Cryptosystem with Correction Value Method

Memperkuat algoritma kriptografi yang lemah dengan metode "correction value"

Habibie Faried
Informatics Engineering
Institut Teknologi Bandung
Bandung, Indonesia
habibiefaried@gmail.com

***Abstract*—This paper talks about how to harden security in classic cyptosystem (e.g vigenere, etc). Those cryptosystem is very weak because generated cipher has predictable distribution character. This paper also proposes new method called "Correction Value", it means that we use some values to randomize the cipher output.**

Keywords—correction value; weak cryptography; vigenere;

I. BACKGROUND

Most of old cyptography algorithm are vulnerable to frequency analysis attack like vigenere cipher, playfair cipher and others. The main problem those algorithms always produce cipher text that has same letter distribution. If that distribution is able to be predicted, then it can be deciphered without passkey.

One of the solution is randomizing letter distribution. So, if we can obfuscate the cipher letter then hacker will not have an idea how to predict plain text. We called that "correction value". It means that generated cipher's letter will be corrected with some value.

This correction value is very simple, only using quadratic equation cipher graph. Graph always produces some values that will be applied to cipher's character. These values depends on quadratic equation. So, it will randomize cipher's output and secure them from *frequency analysis attack.*

II. BASIC THEORY

A. *Vigenere Cipher*

Vigenere Cipher is one of *polyalphabetic substitution cipher*. Published by cryptologist Blaise de Vigenere in 16th Century. Vigenere Cipher is used by Confederate Army in American Civil War. Vigenere Cipher use *vigenere square* to encrypt plain text.

For each rows in *vigenere square* is generated cipher text character based on selected plain text character. In mathematics formulation. Vigenere Cipher encryption is described at below:

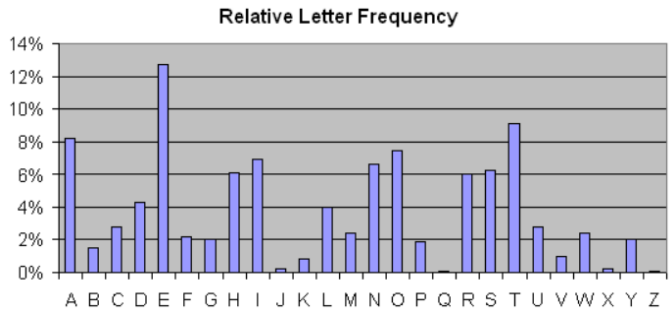
$$C = (p + k)\ mod\ 256$$

- C is Cipher character
- p is plain character
- k is key character

B. *Frequency Analysis Attack*

Frequency Analysis is methodology for breaking weak substitution cipher algorithm. These ciphers replace one letter of the plaintext with another to produce ciphertext. For example, letter A is changed by X.

This method of attack is used to decrypting *polyalphabetic substitution cipher*. Main idea of this attack is relating cipher's character frequency with plain character frequency. For example, english word many letter "e" on it. All of letter frequency is described below



Source: http://en.wikibooks.org/wiki/Cryptography/Frequency_analysis

This method has been improved to break vigenere cipher. This advanced method is called "Kasiski" method. Friedrich Kasiski is the first man who successfully break vigenere cipher. Kasiski method helps us to determine length of the vigenere key.

Document in English not only contains letter repetitions, but also word repetitions such as "THE", "ON", etc. So, these repetition plain words may result repetition cipher words too. The distance of two same words can be a key length multiplication.

III. ALGORITHM DESIGN

Our cases is implementing "correction value" algorithm to Vigenere Cipher. This correction value will use quadratic equation

$$y = ax^2 + bx + c$$

This is an explanation about those variables in equation:
• y is correction value will be set into Vigenere Cipher
• a is constant integer
• b is constant integer
• c is constant integer
• x is plain letter's index. It will be incremented for each of plain letter

Encyption Formula:

$$enc = Vigenere.Encrypt(plain)$$

$$enc = enc + y$$

Decryption Formula:

$$dec = dec - y$$

$$plain = Vigenere.Decrypt(dec)$$

*Note: This Vigenere Cipher is using ASCII Table (character 0 – 255). All operation in integer, output is represented as Hex Value

This correction value will randomize cipher's characters. Because of plain's character index is incremented. So, it will produce different value of y (correction value). For example like calculation sample below

Plain character: 'a'
Passkey : 123
Plain character according to ASCII table: 97
$y = x^2 - x + 2$

because of only 1 character, value of x is 1. So, we will calculate y first:

$y = (1)^2 - 1 + 2$
$y = 1$

Then, calculate the cipher character.
Cipher character is plain character shifted right with passkey and modulo the limit.
In ASCII table, key character "1" is 49. Then,

enc = (97 + 49) mod 256 = 146
enc = enc + y
enc = 146 + 1 = 147

So, generated cipher character is 147. Convert it to hex value and print to file.

IV. TESTING AND ANALYSIS

Vigenere Cipher with Correction Value successfully implemented in Java Language. Now, it must be tested with plain text. So, this is the plaintext

A. Plaintext

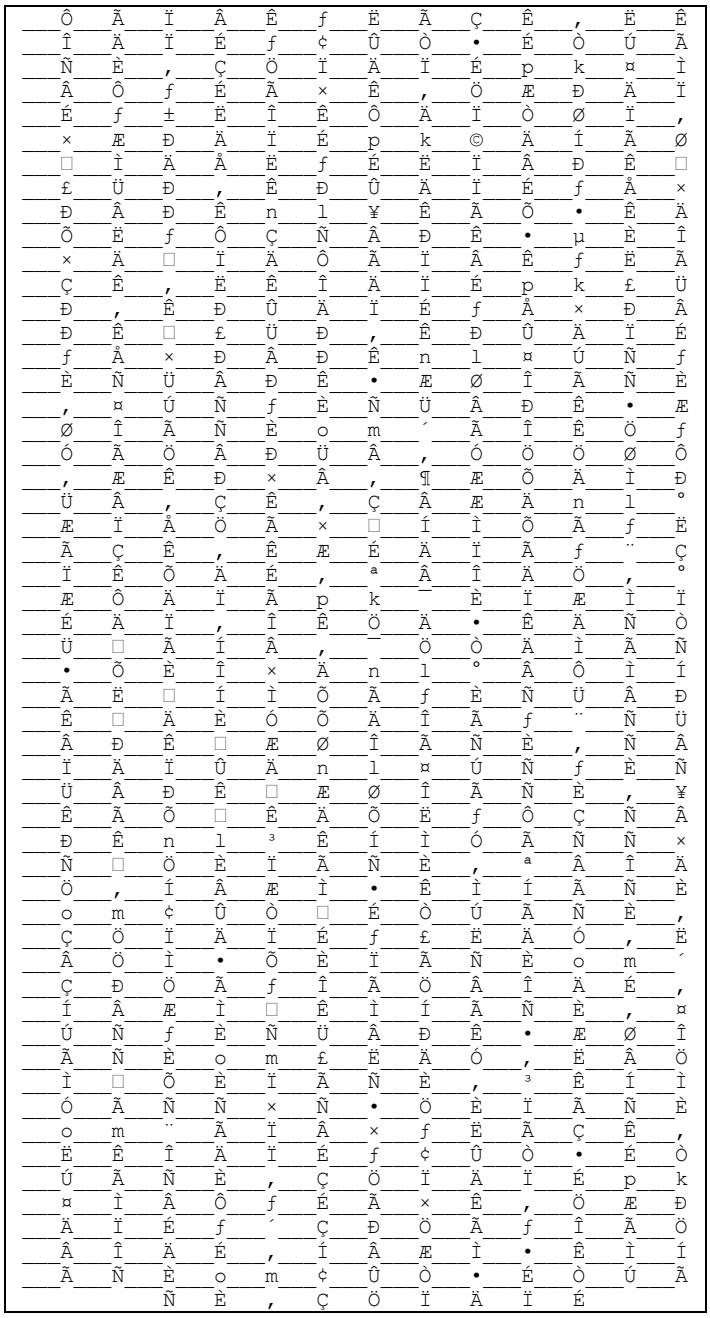
```
Sakit rasanya putus cinta Sesaknya di dada
Membuat kita jadi gegana Gelisah Galau Merana
Mendingan kita happy aja Lupakan semua
Marilah kita goyang bersama Goyang dumang
namanya
Ayo goyang dumang Biar hati senang
Pikiranpun tenang Galau jadi hilang
Ayo goyang dumang Biar hati senang
Semua masalah jadi hilang Ayo goyang dumang
Biar hati senang Pikiranpun tenang
Galau jadi hilang Ayo goyang dumang
Biar hati senang Semua masalah jadi hilang
Ayo goyang dumang Ayo goyang dumang
Ayo goyang dumang Ayo goyang dumang
Sakit rasanya putus cinta Sesaknya di dada
Membuat kita jadi gegana Gelisah Galau Merana
Mendingan kita happy aja Lupakan semua
Marilah kita goyang bersama Goyang dumang
namanya
Ayo goyang dumang Biar hati senang
Pikiranpun tenang Galau jadi hilang
Ayo goyang dumang Biar hati senang
Semua masalah jadi hilang Ayo goyang dumang
Biar hati senang Pikiranpun tenang
Galau jadi hilang Ayo goyang dumang
Biar hati senang Semua masalah jadi hilang
Ayo goyang dumang
```

B. Ciphertext 1

Cipher text will be generated by Vigenere Cipher without correction value. Using original vigenere cipher. Vigenere Passkey is "123"

$$y = 0$$

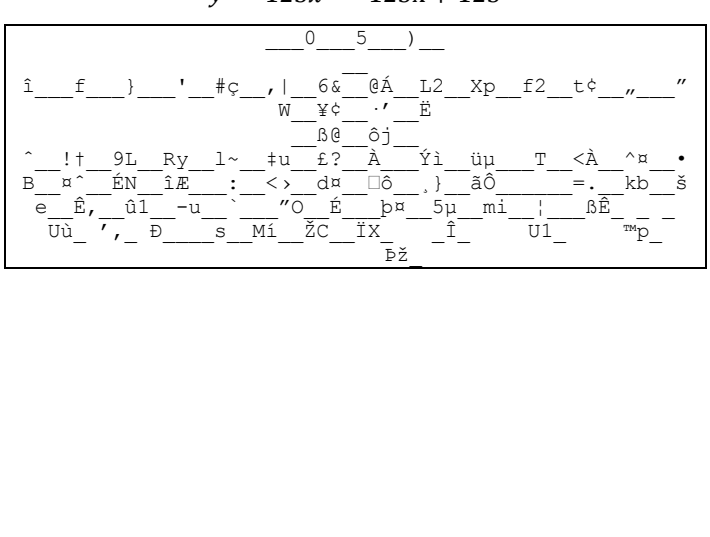
```
 µ Ä Ì Ë × • Ô Ä Ô Ã Ñ Ú Ã
ƒ Ñ × × Ö Õ ƒ Ä Ë Ñ Õ Ã ƒ
´ Ç Ö Â Í Ñ Ú Ã ƒ Å Ë ƒ Å
Ã Ç Â o m ® Ç Ð Ã × Ä Õ ,
Î Ê Ö Ä □ Ì Ä Å Ë ƒ È Ç Ê
Â Ð Ä □ © È Í Ë Ö Â Ê ƒ ¨
Ã Ï Ã × ƒ ® Ç Õ Â Ð Ä n l
° Æ Ð Ç Ê Ð Ê Â Ð ƒ Ì Ë ×
Â , Ë Â Ò Ó Ú , Ä Ë Ã ƒ -
× Ó Â Í Ä Ï , Ö Æ Ï Ø Â o
m ® Ã Õ Ê Î Ä É , Î Ê Ö Ä
□ É Ò Ú Ã Ñ È , Å Æ Ô Ö Â
Ï Ä • © Ò Ú Ã Ñ È , Ç Ö Ï
Ä Ï É ƒ Ï Ã Ð Â Ð Ü Â o m
ç Û Ò □ É Ò Ú Ã Ñ È , Ç Ö
Ï Ä Ï É ƒ £ Ë Ä Ó , Ë Â Ö
Ì • Õ È Ï Ã Ñ È o m ± Ë Î
Ê Ô Ä Ï Ò Ø Ï , × Æ Ð Ä Ï
É ƒ ¨ Ã Ï Â × ƒ Ë Ã Ç Ê ,
Ë Ê Î Ä Ï É p k £ Ü Ð , Ê
Ð Û Ä Ï É ƒ Å × Ð Â Ð Ê •
¤ Ì Ã Ô ƒ É Ã × Ê , Ö Æ Ð
Ä Ï É p k µ È Î × Ä • Ï Ä
```

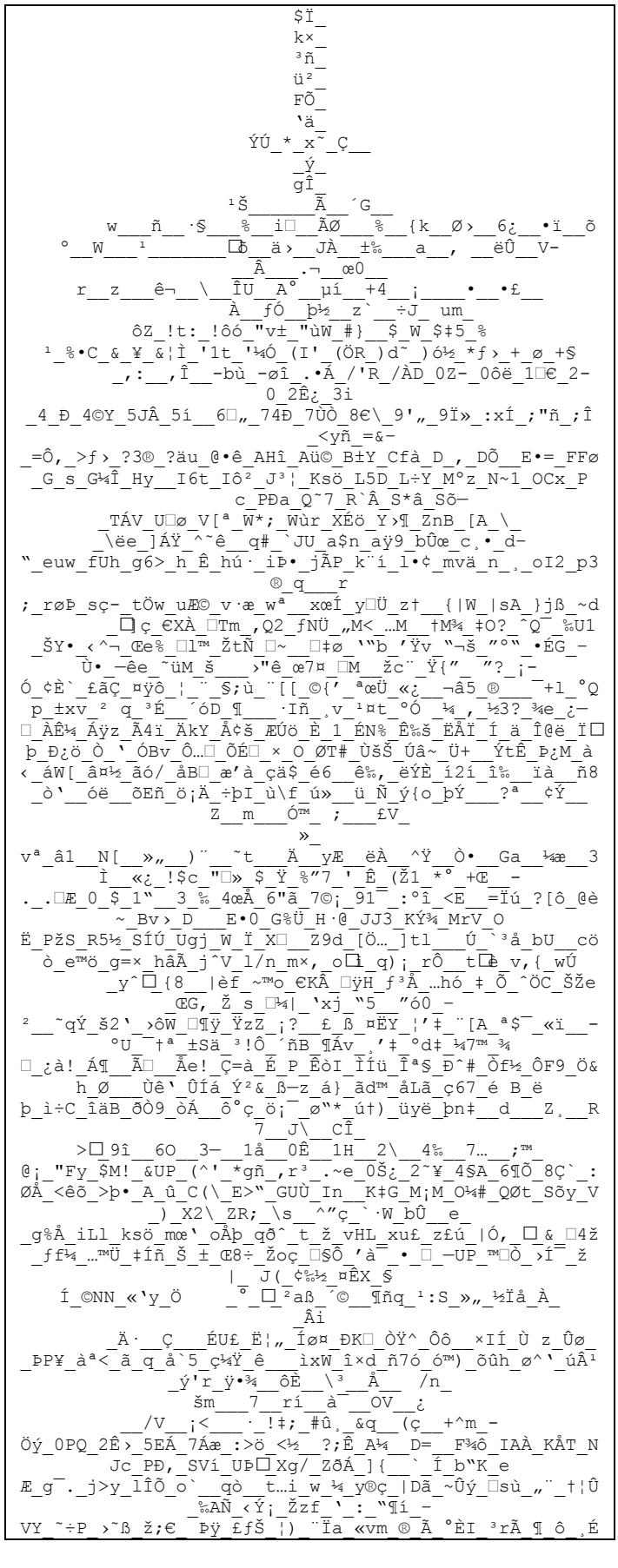


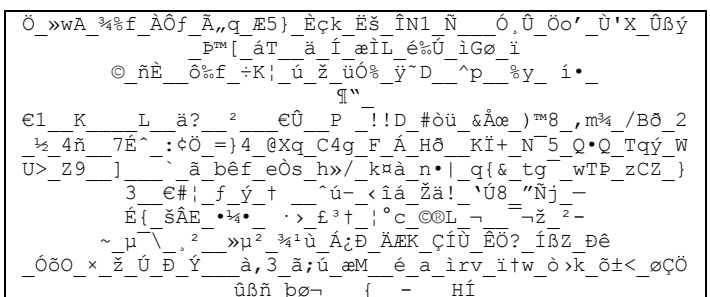
C. Ciphertext 2

Cipher will be generate by vigenere cipher with correction value. Using same passkey and this quadratic equation correction value

$$y = 123x^2 - 123x + 123$$



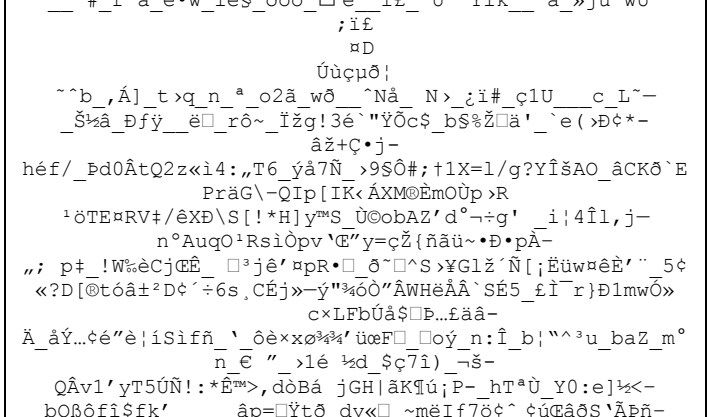




D. Ciphertext 3

Same as ciphertext 2 above. But with different value correction equation. Use the same vigenere passkey

$$y = 250000x^2 - 100000x + 100000$$



```
          q_ª£_nÇDî‟ãî_ö_žðR_Wbî:_v”Vß+is®□_□…xm-
  d_’ÊJIæç8$\_–Ÿ¡!*¼š>/zl[;Ù]xOÙc•kz¡²Ž¼èÏ¹Ÿÿìì$•
          &Jr'h_{D±ybb_,!□[,™œ»wò°#dj×’ñãõ
          ¡_^ðh0_a_M□sgk3&¦^Ðzi¦upTÄ"_ïáÖ>™
```

V. SECURITY ANALYSIS

First cipher text is vulnerable to *frequency analysis attack*. Because it has repetitions and predictable distribution letter. It's too bad because hacker can break this cipher with *known plaintext* only.

Second cipher is using correction value. It has big difference from the first cipher. This cipher is random and cannot be counted by *frequency analysis.* Same as third cipher using bigger parameter. It has more random character distribution, that means it's also safe.

VI. CONCLUSION

- This algorithm is very fast because only addition, substraction and multiplication.
- Breaking this algorithm is very hard. Because, it can't be attacked by frequency analysis attack. It always generates cipher text with very random letter.
- Breaking the passkey by bruteforce attack will be very hard too. Because, hacker has to know at least 4 parameter:
    - Vigenere passkey (string)
    - Variable "a" (integer)
    - Variable "b" (integer)
    - Variable "c" (integer)

For example, if we have only 6 characters passkey ([0-9][A-Z][a-z]). Hacker must generate $(10 + 26 + 26)^6$ all of characters. Then to predict the quadratic curve, hacker must generate value of a,b,c. Integer has range $(-2)^{31} – (2)^{31}$ value. In result, there are $(2^{32})^3$ possible curves.

So, this method is cheap and strong. It can be used to harden the security of weak cryptosystem

REFERENCES

[1] Munir, Rinaldi, Presentation Slide: "Algoritma Kriptografi Klasik_bag2". Bandung: ITB

[2] Munir, Rinaldi, Presentation Slide: "Kriptanalisis". Bandung: ITB