A Modified Playfair Cipher for Encrypting Digital Files

Novriady Saputra 13510083
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13510083@std.stei.itb.ac.id

***Abstract*—In this paper, a new algorithm based on Playfair Cipher is proposed to encrypt digital files. The major change in this algorithm is that the key is modified as the encryption process continues. The key modification is based on previously generated cipher byte. Experiment result shows that a slight change of the key will result in different cipher file. Experiment results also shows that a slight change of the plain file will produce different cipher file.**

***Index Terms*—Playfair Cipher, Symmetric Cryptography**

I. Introduction

The Playfair Cipher is the first digraph-based classic cipher invented by Charles Wheatstone and promoted by Lord Playfair. This cipher uses a 5x5 box as the key; each cell contains character A to Z, excluding character J.

To encrypt a text, all the character J in the text must be replaced with the character I. This is because the key has no character J. The text is then divided into groups of two characters; if any group contains double letters, or only a character is left, insert an extra dummy character (usually X or Z).

After the pre-processing, the text is then encrypted using the key following these rules:

i) If the two characters appear in the same row of the key, replace each character with the character to the right (wrapping) of it in the key matrix.
ii) If the two characters appear in the same column of the key, replace each character with the character to the bottom (wrapping) of it in the key matrix.
iii) If the two characters don't appear in the same row and the same column of the key, replace each character with the character at the other rectangle corner at the same row.

To decrypt a text, the text must be divided into groups of two characters. The text is then decrypted using the key following these rules:

i) If the two characters appear in the same row of the key, replace each character with the character to the right (wrapping) of it in the key matrix.
ii) If the two characters appear in the same column of the key, replace each character with the character to the bottom (wrapping) of it in the key matrix.
iii) If the two characters don't appear in the same row and the same column of the key, replace each character with the character at the other rectangle corner at the same row.

After the decryption process completed, a post-processing is needed. The post-processing includes removing dummy characters and replacing character I with character J when necessary.

This cipher, however, have some weaknesses. Those weaknesses are:

i) The same digraph in plain text always results in same digraph in cipher text. This leads to a possibility of frequency analysis attack.
ii) The classic Playfair Cipher can only be applied to texts. Nowadays, however, not only text needs to be encrypted.

Therefore, the Playfair Cipher needs to be improved, as proposed in this paper.

II. Proposed Algorithm

The proposed algorithm consists of two phases, the encryption algorithm and decryption algorithm. These two sub-algorithms will be explained separately.

A. *Encryption Algorithm*

The proposed algorithm uses a 16x16 substitution box instead of the classical 5x5 box. This substitution box is filled with bytes from 0 to 255, each value occurs once. The starting value of this substitution box will be the key.

The plain file is divided per two bytes. Every two bytes of the plain file is then substituted with a new pair using the current key by following these rules:

i) If there is only one byte left, or if the two bytes are the same, then replace each byte with the value of the key at row x and column y such that $16x + y$ equal to the byte being processed.
ii) If the two bytes are in the same row of the key, replace each byte with the value to the right (wrapping) of it.
iii) If the two bytes are in the same column of the key, replace each byte with the value to the bottom (wrapping) of it.
iv) If the two bytes are neither in the same row nor the same column of the key, replace each byte with the value at the other rectangle corner at the same row.

These rules have slight difference compared to the classical one when handling digraphs with same unit. This is because adding dummy bytes in the encryption process will require removing dummy bytes in the decryption process, which is difficult to do (especially because the dummy bytes can be anywhere).

In addition, after every four bytes are encrypted, the key will be modified using the cipher byte by following these rules:

i) The first two bytes will determine which rows to be modified. For each bit of these bytes (traversed from first to the second byte, and from MSB to LSB), if the bit is 1, right shift (wrapping) the corresponding row one place.

ii) The last two bytes will determine which column to be modified. For each bit of these bytes (traversed from first to the second byte, and from MSB to LSB), if the bit is 1, down shift (wrapping) the corresponding column one place.

B. *Decryption Algorithm*

The proposed algorithm uses a 16x16 substitution box instead of the classical 5x5 box. This substitution box is filled with bytes from 0 to 255, each value occurs once. The starting value of this substitution box will be the key.

The cipher file is divided per two bytes. Every two bytes of the cipher file is then substituted with a new pair using the current key by following these rules:

i) If there is only one byte left, or if the two bytes are the same, then replace each byte with 16x + y such that the value of the key at row x and column y equal to the byte being processed.

ii) If the two bytes are in the same row of the key, replace each byte with the value to the left (wrapping) of it.

iii) If the two bytes are in the same column of the key, replace each byte with the value to the top (wrapping) of it.

iv) If the two bytes are neither in the same row nor the same column of the key, replace each byte with the value at the other rectangle corner at the same row.

In addition, after every four bytes are decrypted, the key will be modified using the plain bytes by following these rules:

i) The first two bytes will determine which rows to be modified. For each bit of these bytes (traversed from first to the second byte, and from MSB to LSB), if the bit is 1, right shift (wrapping) the corresponding row one place.

ii) The last two bytes will determine which column to be modified. For each bit of these bytes (traversed from first to the second byte, and from MSB to LSB), if the bit is 1, down shift (wrapping) the corresponding column one place.

III. Experiment Result and Discussion

An experiment has been conducted to test the algorithm. Several basic security tests have also been done.

A. *Encryption and Decryption Result*

The test uses a pdf file as the plain text. The preview of the pdf file is shown by Fig. 1.

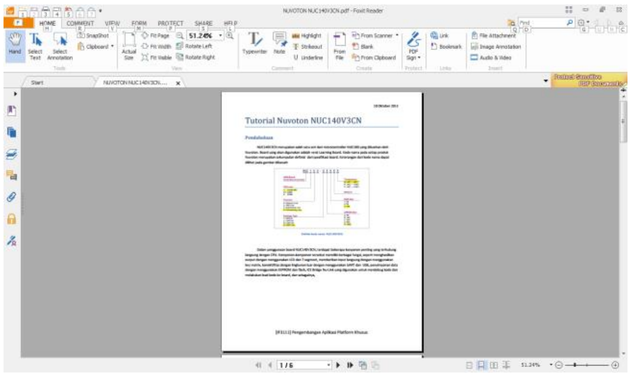
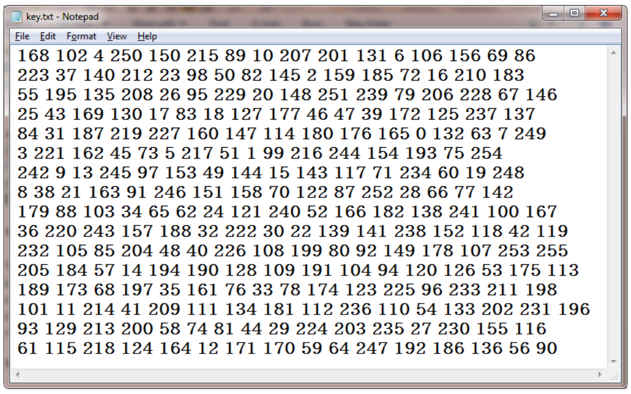

NUVOTON NUC140V3CN.pdf - Foxit Reader
HOME
COMMENT
VIEW
FORM
PROTECT
SHARE
HELP
Tutorial Nuvoton NUC140V3CN
51.24%
1 / 6

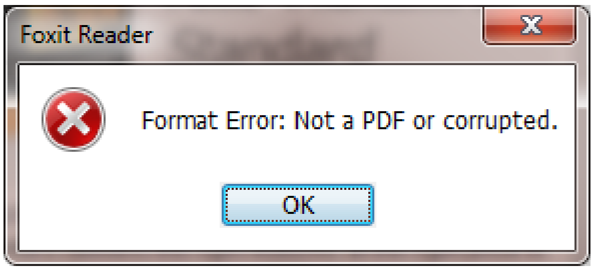
Fig. 1 - Preview of the Plain File

This file is then encrypted with the key, as shown in Fig. 2, producing a pdf file that cannot be opened, as shown in Fig. 3.



```
168 102 4 250 150 215 89 10 207 201 131 6 106 156 69 86
223 37 140 212 23 98 50 82 145 2 159 185 72 16 210 183
55 195 135 208 26 95 229 20 148 251 239 79 206 228 67 146
25 43 169 130 17 83 18 127 177 46 47 39 172 125 237 137
84 31 187 219 227 160 147 114 180 176 165 0 132 63 7 249
3 221 162 45 73 5 217 51 1 99 216 244 154 193 75 254
242 9 13 245 97 153 49 144 15 143 117 71 234 60 19 248
8 38 21 163 91 246 151 158 70 122 87 252 28 66 77 142
179 88 103 34 65 62 24 121 240 52 166 182 138 241 100 167
36 220 243 157 188 32 222 30 22 139 141 238 152 118 42 119
232 105 85 204 48 40 226 108 199 80 92 149 178 107 253 255
205 184 57 14 194 190 128 109 191 104 94 120 126 53 175 113
189 173 68 197 35 161 76 33 78 174 123 225 96 233 211 198
101 11 214 41 209 111 134 181 112 236 110 54 133 202 231 196
93 129 213 200 58 74 81 44 29 224 203 235 27 230 155 116
61 115 218 124 164 12 171 170 59 64 247 192 186 136 56 90
```

Fig. 2 - Key Used for Encryption



Format Error: Not a PDF or corrupted.

Fig. 3 - Foxit Reader Cannot Open Cipher File

The encrypted file is then decrypted with the same key, resulting in a readable pdf file, exactly same as the plain file, as shown in Fig. 4.

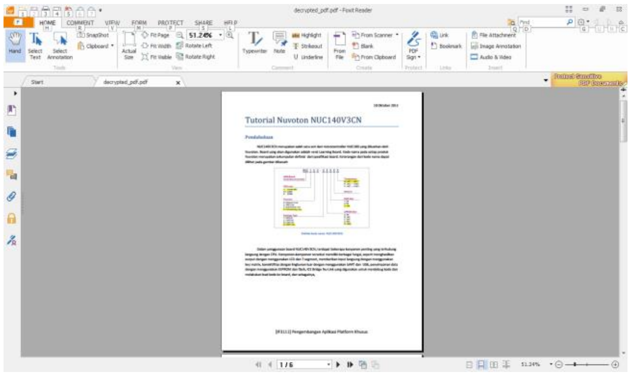


Fig. 4 – Preview of the Decrypted File

This experiment result shows that the proposed algorithm successfully encrypted the plain file. The proposed algorithm also successfully decrypted the cipher file. Thus the algorithm worked properly.

B. *Key Space Analysis*

The proposed algorithm uses a 16x16 box as the key, with each cell contains different byte. The first cell has 256 possibilities. Because the a byte can only be used once, the second cell has only 255 possibilities. This continues on until the last cell, giving a total of 256! possibilities of a key. The large total possible key combinations make the brute-force attack is impossible to do.

C. *Byte Histogram Analysis*

Byte histogram of a file is a visualization of byte distribution in a file. Byte histogram is one of important aspect in cryptography, because by analyzing histogram, an attacker can find the connection between plain file and cipher file. Attempting to find connection between plain file and cipher file using byte histogram is known as *frequency analysis attack*.

The byte histogram for the plain file is shown by Fig. 5, whereas the byte histogram for the cipher file is shown by Fig. 6.

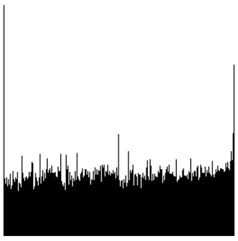


Fig. 5 - Byte Histogram for Plain File

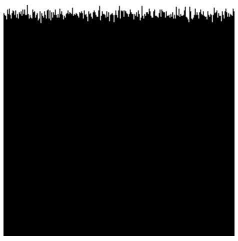
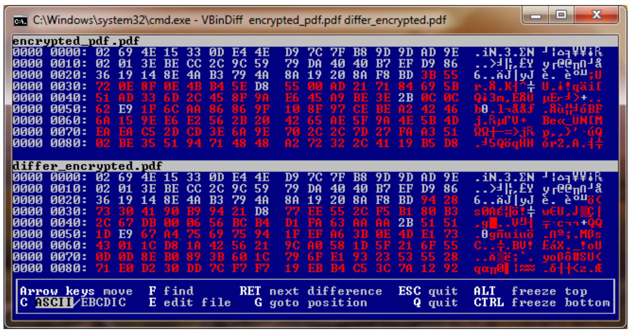


Fig. 6 - Byte Histogram for Cipher File

A good encryption algorithm should presents cipher file such that connection between the plain file and cipher file cannot be obtained by analyzing the byte histogram of these files. In this experiment result, the byte histogram of the plain file shows a great occurrence of byte 0 and byte 255, whereas the byte histogram of the cipher file shows almost even occurrence of all bytes. This implies that frequency analysis attack is very difficult to be done.

D. *Key Sensitivity Test*

A key's sensitivity is how much the cipher file differs when the key is changed slightly; the greater difference, the more sensitive a key is. A sensitive key is important, because an insensitive key can aid attacker to retrieve the correct key.

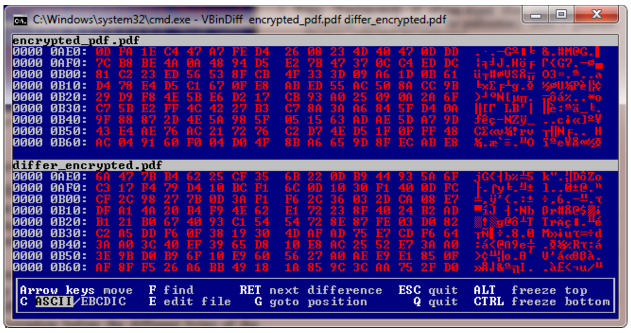


```
C:\Windows\system32\cmd.exe - VBinDiff  encrypted_pdf.pdf differ_encrypted.pdf
encrypted_pdf.pdf
0000 0000: 02 69 4E 15 33 0D E4 4E  D9 7C 7F B8 9D 9D AD 9E
0000 0010: 02 01 3E BE CC 2C 9C 59  79 DA 40 40 B7 EF D9 86
0000 0020: 36 19 14 8E 4A B3 79 4A  8A 19 20 8A F8 BD 3B 55
0000 0030: 72 0E 8F 0E 4B B4 5E D8  55 00 AD 21 71 84 69 5B
0000 0040: 51 AD 33 6D 2C 45 8F 9A  E6 45 A9 BE 3E 2B 0C 0C
0000 0050: 62 E9 1F 6C AA 86 86 9F  10 8F 97 CE BE A2 42 46
0000 0060: 6A 15 9E E6 E2 56 2B 20  42 65 AE 5F 9A 4E 5B 4D
0000 0070: EA EA C5 2D CD 3E 6A 9E  70 2C 2C 7D 27 FA A3 51
0000 0080: 02 BE 35 51 94 71 48 48  A2 72 32 2C 41 19 B5 D8

differ_encrypted.pdf
0000 0000: 02 69 4E 15 33 0D E4 4E  D9 7C 7F B8 9D 9D AD 9E
0000 0010: 02 01 3E BE CC 2C 9C 59  79 DA 40 40 B7 EF D9 86
0000 0020: 36 19 14 8E 4A B3 79 4A  8A 19 20 8A F8 BD 94 28
0000 0030: 73 30 41 90 B9 94 21 D8  77 EE 55 2C F5 B1 80 B3
0000 0040: 2C 67 DB 00 06 56 BC B4  D1 FA 63 AA AA 2B 51 51
0000 0050: 1D E9 67 A4 75 69 75 94  1F EF A6 3B 0E 4D E1 73
0000 0060: 43 01 1C D8 1A 42 56 21  9C A0 58 1D 5F 21 6F 55
0000 0070: 0D 0D 8E B0 89 3B 60 1C  79 6F E1 93 23 53 55 28
0000 0080: 71 E0 D2 30 DD 7C F7 F7  19 EB B4 C5 3C 7A 12 92

Arrow keys move   F find        RET next difference   ESC quit   ALT  freeze top
C ASCII/EBCDIC    E edit file     G goto position       Q quit   CTRL freeze bottom
```

Fig. 7 - Comparation Between Cipher Files (Beginning)



```
C:\Windows\system32\cmd.exe - VBinDiff  encrypted_pdf.pdf differ_encrypted.pdf
encrypted_pdf.pdf
0000 0AE0: 0D FA 1E C4 47 A7 FE D4  26 08 23 4D 40 47 0D DD
0000 0AF0: 7C B8 BE 4A 0A 48 94 D5  E2 7B 47 37 0C C4 ED DC
0000 0B00: 81 C2 23 ED 56 53 8F CB  4F 33 3D 09 A6 1D 0B 61
0000 0B10: D4 78 E4 D5 C1 67 0F E8  AB ED 55 AC 50 8A CC 9B
0000 0B20: 29 D9 F8 4E 5B E6 D2 17  CB 93 A0 25 09 0A 2A 6F
0000 0B30: C7 5B E2 FF 4C 42 27 B3  C7 8A 3A A6 84 5F D4 0A
0000 0B40: 9F 88 87 2D 4E 5A 98 5F  05 15 63 AD AE 5D A7 9D
0000 0B50: 43 E4 AE 76 AC 21 72 76  C2 D7 4E D5 1F 0F FF 48
0000 0B60: AC 04 91 60 F0 04 D0 4F  8B A6 65 9D 8F EC AB E8

differ_encrypted.pdf
0000 0AE0: 6A 47 7B B4 62 25 CF 35  6B 22 0D B9 44 93 5A 6F
0000 0AF0: C3 17 F4 79 D4 10 BC F1  6C 0D 10 30 F1 40 0D FC
0000 0B00: CF 2C 98 27 7B 0D 3A F1  F6 2C 36 03 2D CA 08 E7
0000 0B10: DF A1 4A 20 B4 F9 4E 62  E1 72 23 8F 40 24 B2 AD
0000 0B20: B1 21 B0 67 40 93 C1 54  54 72 8E 87 FE 03 D0 82
0000 0B30: C2 A5 DD F6 0F 38 19 30  4D AF AD 75 E7 CD F6 64
0000 0B40: 3A A0 3C 40 EF 39 65 D8  10 E8 AC 25 52 E7 3A A0
0000 0B50: 3E 9B D0 B9 6F 10 E9 60  56 27 A0 AE E9 E1 85 0F
0000 0B60: AF 8F F5 26 A6 BB 49 18  1A 85 9C 3C AA 75 2F D0

Arrow keys move   F find        RET next difference   ESC quit   ALT  freeze top
C ASCII/EBCDIC    E edit file     G goto position       Q quit   CTRL freeze bottom
```

Fig. 8 - Comparation Betwen Cipher Files (Middle)

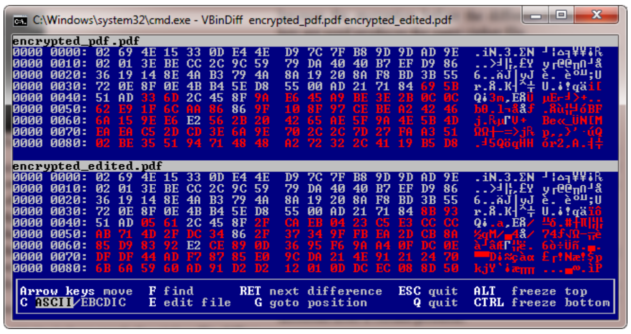
The plain file is encrypted twice with different keys. The key used for the second encryption is obtained by swapping the position of byte 99 and byte 115. Fig.7 shows the comparation between the two cipher files at the beginning of the file, whereas Fig. 8 shows the comparation between the two cipher files at the middle of the file. These comparisons are made with a program called VBinDiff.exe.

In this experiment result, the plain file and cipher file have similarity at the beginning of the file, but are completely different after a certain position. This is because the encryption before the different bytes of the key are used for the first time produces the same cipher file.

Although there are similarities at the beginning of the file, the entire body of the files are still very different. This means attempting to retrieve the correct key by trying various key is very difficult to do.

E. *Diffusion Test*

Diffusion in cryptography means that a slight difference in plain text should affect the cipher text significantly. Encryption algorithm that is not diffused can be cracked easily if the plain text is known. This kind of attack is known as *known plain text attack*.

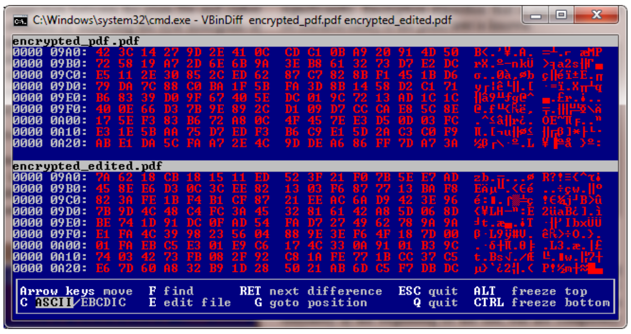


```
C:\Windows\system32\cmd.exe - VBinDiff  encrypted_pdf.pdf encrypted_edited.pdf
encrypted_pdf.pdf
0000 0000: 02 69 4E 15 33 0D E4 4E  D9 7C 7F B8 9D 9D AD 9E  .iN.3.ΣN ┘|⌂╕¥¥¡₧
0000 0010: 02 01 3E BE CC 2C 9C 59  79 DA 40 40 B7 EF D9 86  ..>╛╠,£Y y┌@@╖∩┘å
0000 0020: 36 19 14 8E 4A B3 79 4A  8A 19 20 8A F8 BD 3B 55  6..ÄJ│yJ è. è°╜;U
0000 0030: 72 0E 8F 0E 4B B4 5E D8  55 00 AD 21 71 84 69 5B  r.Å.K┤^╪ U.¡!qäi[
0000 0040: 51 AD 33 6D 2C 45 8F 9A  E6 45 A9 BE 3E 2B 0C 0C  Q¡3m,EÅÜ µE⌐╛>+..
0000 0050: 62 E9 1F 6C AA 86 86 9F  10 8F 97 CE BE A2 42 46  bΘ.l¬ååƒ .Åù╬╛óBF
0000 0060: 6A 15 9E E6 E2 56 2B 20  42 65 AE 5F 9A 4E 5B 4D  j.₧µΓV+  Be«_ÜN[M
0000 0070: EA EA C5 2D CD 3E 6A 9E  70 2C 2C 7D 27 FA A3 51  ΩΩ┼-═>j₧ p,,}'·úQ
0000 0080: 02 BE 35 51 94 71 48 48  A2 72 32 2C 41 19 B5 D8  .╛5QöqHH ór2,A.╡╪

encrypted_edited.pdf
0000 0000: 02 69 4E 15 33 0D E4 4E  D9 7C 7F B8 9D 9D AD 9E  .iN.3.ΣN ┘|⌂╕¥¥¡₧
0000 0010: 02 01 3E BE CC 2C 9C 59  79 DA 40 40 B7 EF D9 86  ..>╛╠,£Y y┌@@╖∩┘å
0000 0020: 36 19 14 8E 4A B3 79 4A  8A 19 20 8A F8 BD 3B 55  6..ÄJ│yJ è. è°╜;U
0000 0030: 72 0E 8F 0E 4B B4 5E D8  55 00 AD 21 71 84 8B 93  r.Å.K┤^╪ U.¡!qäïô
0000 0040: 51 AD 05 61 2C 45 8F 2F  CA EB 04 23 C5 E3 CC CC  Q¡.a,EÅ/ ╩δ.#┼π╠╠
0000 0050: AB 71 4D 2F DC 34 86 2F  37 34 9F FB EA 2D CB 8A  ½qM/▄4å/ 74ƒ√Ω-╦è
0000 0060: 85 D9 83 92 E2 CE 89 0D  36 95 F6 9A A4 0F DC 0E  à┘âÆΓ╬ë. 6ò÷Üñ.▄.
0000 0070: DF DF 44 AD F7 87 85 E0  9C DA 21 4E 91 21 24 70  ▀▀D¡≈çàα £┌!Næ!$p
0000 0080: 6B 6A 59 60 AD 91 D2 D2  12 01 0D DC EC 08 8D 50  kjY`¡æ╥╥ ...▄∞.ìP

 Arrow keys move   F find       RET next difference   ESC quit   ALT  freeze top
 C ASCII/EBCDIC    E edit file    G goto position       Q quit   CTRL freeze bottom
```

Fig. 9 - Comparison Between Cipher Files (Beginning)



```
C:\Windows\system32\cmd.exe - VBinDiff  encrypted_pdf.pdf encrypted_edited.pdf
encrypted_pdf.pdf
0000 09A0: 42 3C 14 27 9D 2E 41 0C  CD C1 0B A9 20 91 4D 50  B<.'¥.A. ═┴.⌐ æMP
0000 09B0: 72 58 19 A7 2D 6E 6B 9A  3E B8 61 32 73 D7 E2 DC  rX.º-nkÜ >╕a2s╫Γ▄
0000 09C0: E5 11 2E 30 85 2C ED 62  87 C7 82 8B F1 45 1B D6  σ..0à,φb ç╟éï±E.╓
0000 09D0: 79 DA 7C 88 C0 BA 1F 5B  FA 3D 8B 14 58 D2 C1 71  y┌|ê└║.[ ·=ï.X╥┴q
0000 09E0: B6 83 39 D0 9F 67 40 5E  DC 01 9C 72 13 AD 1C 1C  ╢â9╨ƒg@^ ▄.£r.¡..
0000 09F0: 40 0E 66 D3 7B 9E 89 2C  D1 09 D7 CC CA E8 5C 8E  @.f╙{₧ë, ╤.╫╠╩Φ\Ä
0000 0A00: 17 5E F3 83 B6 72 A8 0C  4F 45 7E E3 D5 0D 03 FC  .^≤â╢r¿. OE~π╒..ⁿ
0000 0A10: E3 1E 5B AA 75 D7 ED F3  B6 C9 E1 5D 2A C3 C0 F9  π.[¬u╫φ≤ ╢╔ß]*├└∙
0000 0A20: AB E1 DA 5C FA A7 2E 4C  9D DE A6 86 FF 7D A7 3A  ½ß┌\·º.L ¥▐ªå }º:

encrypted_edited.pdf
0000 09A0: 7A 62 18 CB 18 15 11 ED  52 3F 21 F0 7B 5E E7 AD  zb.╦...φ R?!≡{^τ¡
0000 09B0: 45 8E E6 D3 0C 3C EE 82  13 03 F6 87 77 13 BA F8  EÄµ╙.<εé ..÷çw.║°
0000 09C0: 82 3A FE 1B F4 B1 CF 87  21 EE AC 6A D9 42 3E 96  é:■.⌠▒╧ç !ε¼j┘B>û
0000 09D0: 7B 9D 4C 48 C4 FC 3A 45  32 81 61 42 A8 5D 06 8D  {¥LH─ⁿ:E 2üaB¿].ì
0000 09E0: BE 74 1D 91 DC 0F AD 54  FA D7 27 49 62 78 9A 9A  ╛t.æ▄.¡T ·╫'IbxÜÜ
0000 09F0: E1 FA 4C 39 98 23 56 04  88 9E 3E F6 4F 18 7D 00  ß·L9ÿ#V. ê₧>÷O.}.
0000 0A00: 01 FA EB C5 E3 01 E9 C6  17 4C 33 0A 91 01 B3 9C  .·δ┼π.Θ╞ .L3.æ.│£
0000 0A10: 74 03 42 73 FB 08 2F 92  C8 1A FE 77 1B CC 37 C5  t.Bs√./Æ ╚.■w.╠7┼
0000 0A20: E6 7D 60 A8 32 B9 1D 28  50 21 AB 6D C5 F7 DB DC  µ}`¿2╣.( P!½m┼≈█▄

 Arrow keys move   F find       RET next difference   ESC quit   ALT  freeze top
 C ASCII/EBCDIC    E edit file    G goto position       Q quit   CTRL freeze bottom
```

Fig. 10 - Comparison Between Cipher Files (Middle)

The plain file is modified by editing 1 byte, and then is encrypted using the same key. Fig.9 shows the comparation between the two cipher files at the beginning of the file, whereas Fig. 10 shows the comparation between the two cipher files at the middle of the file. These comparisons are made with a program called VBinDiff.exe.

In this experiment result, the plain file and cipher file have similarity at the beginning of the file, but are completely different after a certain position. This is because the encryption before the different byte of the plain file is encrypted produces the same cipher file. This implies that the algorithm is diffused well; the algorithm can resist known plain text attack.

IV. CONCLUSION

The proposed algorithm performs well; it successfully encrypts and decrypts files. It has a large amount of key space which counters brute-force attack. It also has good encryption algorithm characteristic, such as hiding the connection between plain file and cipher file, diffusing the change of plain file into cipher file, which means resistance against several attacks, such as frequency analysis attack and known-plaintext attack.

REFERENCES

Safwat Hamad et al, *A Modified Playfair Cipher for Encrypting Digital Images*, Modern Science, 2013.

W. Stallings, *Cryptography and Network Security: Principles and Practice 5th edition*, Pearson, 2011

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 19 Maret 2014

[illegible]

Novriady Saputra
13510083