Encryption Binary Images by Using Template Matching

Sundus Khaleel Ebraheem
sunduskhaleel_2019@uomosul.edu.iq
College of Computer Sciences and Mathematics
University of Mosul, Iraq

Received on: 12/09/2005 **Accepted on: 22/01/2006**

ABSTRACT

Text encryption is a very important field in application of data transformation through the digital networks, and the Internet, so it is very necessary to do encryption operation on the text data to get more security in data transformation.

In this paper, we present a method -Template Matching- to encrypt data which is represented in form of image with BMP extension by using Mono Digital Images method with partial compression for the data by using RLE method which increases the security of the method and reduces the file size.

The application results is efficient for the printed or handwritten text in Arabic or English or any other language, and for the maps or sketches images. The method gives a good ability for data encryption. It is suitable for data transformation through the Internet networks.

Keywords: Image Processing, Object Recognition, Images Encryption, Template Matching, Data Encryption, BMP, Mono Images, Data Compression.

تشفير الصور الثنائية باستخدام مطابقة النماذج

سندس خليل إبراهيم

كلية علوم الحاسبات والرياضيات، جامعة الموصل

تاريخ استلام البحث:2005/9/12 **تاريخ قبول البحث:2006/1/22**

الملخص

يعد تشفير النصوص حقلاً مهماً جداً في تطبيقات نقل المعلومات خلال شبكات الاتصالات الرقمية (Digital Networks)، وشبكات الإنترنت، فقد أصبح من الضروري إجراء عمليات التشفير على بيانات النص للحصول على سرية اكثر في النقل .

تم في هذا البحث تقديم طريقة – مطابقة النماذج – لتشفير البيانات الممثلة بشكل صور ذات الامتداد (BMP) باستخدام أسلوب معالجة الصور الرقمية الأحادية (Mono) مع كبس جزئي للبيانات باستخدام طريقة RLE مما زاد من سرية الطريقة وقلل من حجم الرسالة.

وكانت النتائج كفوءة على الرسائل النصية المطبوعة أو المكتوبة باليد باللغة العربية أو الإنكليزية أو أية لغة أخرى فضلا عن صور الرسومات والمخططات، وأعطت الطريقة إمكانية جيدة في تشفير الرسائل، وهي تصلح لنقل البيانات عبر شبكة الاتصالات الإنترنت (Internet).

الكلمات المفتاحية: معالجة الصور، تمييز الكائن، تشفير الصور، مطابقة النماذج ، تشفير البيانات ، BMP ، الصور الاحادية، كبس البيانات ، تشفير الرسائل.

المقدمة

تعد عملية التعامل مع المعلومات في بعض الأحيان مشكلة كبيرة خصوصاً بالنسبة إلى نقل وخزن المعلومات المهمة والتي تحتاج إلى السرية التامة مثل المعلومات العسكرية إذ انه من السهل اختراق نظام الاتصالات دون انكشاف ، لذا يكون الاهتمام في توفير سرية (أمن) البيانات في هذا المجال خاصة اكبر منه في المجالات الأخر[8].

إن تهديدات أمن الاتصالات التي تحيط بالقنوات الاتصالية يمكن التخلص منها باستخدام طرائق التشفير التي تحمي اتصالات البيانات المرسلة من الإفشاء [5].

تقسم طرائق التشفير الحديثة تبعاً لنوع المفتاح المستخدم إلى صنفين الأول يعرف بنظام التشفير ذي المفتاح السري (Secret Key) وهو على نوعين [التشفير الانسيابي والتشفير ألكتلي] [10] ، أو ما يدعى بأنظمة التشفير المتماثلة(Symmetric cipher system)[6,2,1]، والثاني نظام التشفير ذو المفتاح العام (Public Key) [10] ويدعى بأنظمة التشفير غير المتماثلة(Asymmetric cipher system)[6,2,1]، والفرق الأساسي بين النظامين هو في مسألة توزيع المفتاح ، والحل التقليدي لمشكلة توزيع المفتاح السري هو إرساله عبر قناة سرية [10]، وهذه القناة لا يمكن استخدامها في إرسال النصوص (الرسائل) بسبب بطئها وغلاء أسعارها [8]. أما نظام التشفير ذو المفتاح العام فلم يأتِ تعويضاً عن خوارزميات نظام التشفير ذات المفتاح السري، إذ أنها لا تستخدم لغرض تشفير الرسائل بل تستخدم لتشفير المفاتيح، وهي مهمة جداً لمعالجة مشكلة توزيع المفاتيح [3 ,5 ,10 ,12]. ويمكن الإطلاع على مشاكل هذه الطرائق في المصدر [8] والذي أشارت فيه الباحثة إلى مشاكل هذه الطرائق وذكرت المصادر المهمة لذلك.كذلك هنالك أنظمة التشفير التي تدمج بين الطريقين (Mixture of both)[6 ,11].

وبما أن حقل معالجة الصور الرقمية من الحقول العلمية المهمة التي تأخذ دورها بشكل فعال في التطبيقات المختلفة، إذ أن مرحلة الترقيم الممثلة من خلال جهاز الماسح الضوئي(scanner) أو من خلال البرامجيات الخاصة بمعالجة الصور والتحويل بينها، مما يجعل هذه المرحلة كفوءة باستخدام ما متوافر من تكنولوجيا رخيصة[4]. بذلك أصبحت عملية معالجة

الصور بالطريقة الرقمية عملية قليلة الكلفة نسبياً وسهلة التنفيذ وذلك للتطور الحاصل في مجال الحاسوب وأجهزة تحويل وتسجيل محتويات الصور دون فقدان في المعلومات التي تحويها هذه الصور [7].

لذا ارتأينا في هذا البحث أن تكون عملية تشفير النصوص باستخدام الصور النصية أو صور المخططات عن طريق معالجة الصور الرقمية التي تكون مشفرة ضمنيا عند تحويلها إلى بيانات رقمية مخزونة في ملف قراءة وعليه تم اقتراح طريقة لزيادة أمنية الطريقة باستخدام النماذج(الأقنعة) الرقمية في معالجة الصورة .

الطريقة المقترحة

طريقة تكوين الجدول وتحديد رموز بيانات النماذج :

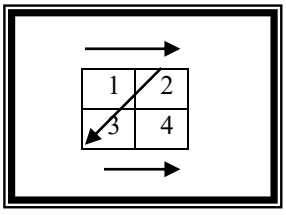
1. يتم تكوين رموز البيانات كما يأتي:

تكوين جميع احتمالات نماذج الفحص كما مبين في الجدول (1)، على فرض ان حجم كل نموذج 2*2 والذي يتفق عليه مسبقا.

الجدول(1) يضم احتمالات نماذج الفحص.

1	0	1	1	1	1	1	0	0	1	0	0
0	0	1	0	0	1	1	1	0	0	0	0
1	1	1	0	0	0	0	0	0	0	0	1
0	0	0	1	1	0	1	0	1	0	0	0
				1	1	1	0	0	1	0	1
				1	1	1	0	0	1	1	0

2. تثبيت تسلسل مواقع معاملات النموذج كأن تكون بالشكل الآتي مثلاً:



الشكل (1) يبين اتجاه تسلسل معاملات النموذج

3. تُكتب بيانات كل نموذج بالتسلسل وذلك بترتيب قيم المعاملات بالتسلسل ابتداءً من معامل النموذج(1) وحتى معامل النموذج (4) حسب الشكل(1).

مثال : النماذج أدناه تكتب بالشكل الأتي على الترتيب (1000 ، 1001 ، 1101). انظر الجدول (2) حقل طريقة كتابة النموذج.

1	1	1	0	1	0
0	1	0	1	0	0

4. نقوم بجمع الواحدات الموجودة في بيانات كل نموذج.

مثال :مجموع بيانات النموذج (1000) = 1

مجموع بيانات النموذج (1001) = 2

مجموع بيانات النموذج (1101) = 3

ستظهر لدينا مجموعة من النماذج يكون فيها مجموع الواحدات متساوياً. وهذا الرقم (المجموع) يمثل أول جزء من رمز البيانات. لاحظ الجدول (2) حقل مجموع الواحدات.

5. نعمل على فصل كل مجموعة على حدة (أي كل مجموعة يكون فيها عدد الواحدات متساوياً).كما في الجدول (2).

6. نرتب الأرقام الثنائية لنماذج كل مجموعة تصاعدياً (عدا بيانات 0 و 4 لأنها تظهر مرة واحدة فقط ولا توجد بيانات مساوية لها بمجموع الواحدات).

مثال: تمثل الأعداد الثنائية الأتية قيم معاملات النماذج التي مجموع الواحدات فيها يساوي(2)، وهي مرتبة تصاعدياً.

1100, 1010, 1001, 0110, 0101,0011

نعطي رمزاً لكل قيمة بالتسلسل ابتداءً بـ (1) وحتى نهاية الأعداد، أي في المثال أعلاه سنعطي التسلسل (1) للعدد (0011) والتسلسل (2) للعدد (0101) ...وهكذا إلى التسلسل (6) للعدد (1100)،هذا التسلسل يمثل الجزء الثاني من رمز البيانات.

7. نُكّون رموز البيانات حيث يُمثَّل كل رمز بجزأين الأول يمثِل مجموع الواحدات في النموذج والجزء الثاني يمثل التسلسل التصاعدي لمجموع معاملات النموذج (الذي تم تكوينه في الخطوة 6). أي المثال أعلاه سيُمثَل بالشكل الأتي:

26, 25, 24, 23, 22, 21

انظر الجدول (2) حقل رمز البيانات.

8. تعاد الخطوة (6،7) على كل مجموعة من النماذج ماعدا النماذج التي مجموع الواحدات فيها يساوي (0)و(4)، حيث يتكون رمز البيانات فيها من جزء واحد فقط وهو مجموع الواحدات الذي

يعطي سرية اكثر للطريقة (لان البيانات 0000 و 1111 لا تتكرر بأشكال أخرى كما ذكرنا أعلاه) وان وضع الناتج بهذه الطريقة يزيد من صعوبة تخمين الخوارزمية من قِبل المتطفل ويطيل من الزمن اللازم لكسر الشفرة.

وبذلك تم تكوين نماذج الفحص المبينة في الجدول(2) الخاصة بعملية التشفير أو فك الشفرة.

الجدول(2) يضم نماذج الفحص وكيفية كتابتها .

	أ	ب	ج	د	هـ	و
	0 0 0 0	1 1 1 1	0 1 1 1	1 0 1 1	1 1 0 1	1 1 1 0
طريقة كتابة النموذج	0000	1111	0111	1011	1101	1110
مجموع الواحدات	0	4	3	3	3	3
رمز البيانات	0	4	31	32	33	34

	ز	ح	ط	ي	ك	ل
	0 0 1 1	0 1 0 1	0 1 1 0	1 0 0 1	1 0 1 0	1 1 0 0
طريقة كتابة النموذج	0011	0101	0110	1001	1010	1100
مجموع الواحدات	2	2	2	2	2	2
رمز البيانات	21	22	23	24	25	26

	م	ن	س	ع
	0 0 0 1	0 0 1 0	0 1 0 0	1 0 0 0
طريقة كتابة النموذج	0001	0010	0100	1000
مجموع الواحدات	1	1	1	1
رمز البيانات	11	12	13	14

مراحل التشفير بالطريقة المقترحة

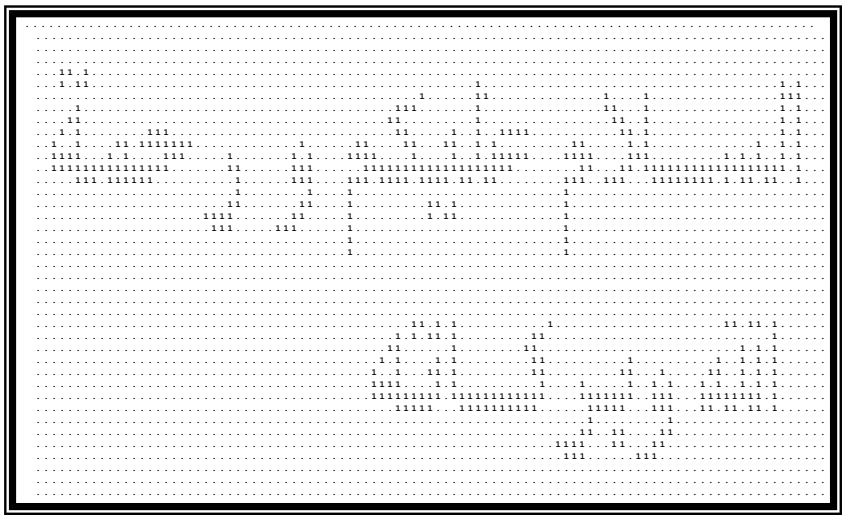
تعتمد طريقة التشفير المقترحة على طرائق معالجة الصورة، وتتميز عن طرائق التشفير الأخرى بان النص أو الرسالة المراد تشفيرها يجب ان تؤخذ كصورة ثنائية ثم تُشَفر بيانات الصورة الثنائية كما موضح لاحقاً وحسب المراحل الأتية:

المرحلة الأولى:

1. يتم خزن الرسالة (الصورة) على هيئة ملف صوري أحادية (Mono) ذي الامتداد (BMP)، ثم نحول بعدها الصورة إلى الصيغة الثنائية. والشكل (2) يمثل نصاً مكتوباً بصيغة ثنائية .

2. تصميم النماذج الخاصة بعملية التشفير (بالاتفاق بين الطرفين) ولتكن بحجم2*2.ثم تحديد رموز البيانات كما موضح في فقرة تكوين الجدول وتحديد رموز بيانات النماذج.

3. إضافة صف من الأصفار أو الواحدات (حسب لون الخلفية إن كان 0 أو 1 على التوالي) إلى الصورة عندما يكون معامل الطول في الصورة فردي، وإضافة عمود من الأصفار أو الوحدات إذا كان معامل العرض في الصورة فردي .(هذا يساعد في عملية مطابقة النماذج لان أبعاد النموذج أرقام زوجية (2*2)).



الشكل(2) صورة نصية بالصيغة الثنائية

المرحلة الثانية :

في هذه المرحلة يتم تشفير الصورة بإجراء تطابق النماذج على الصورة (ابتداءً من الزاوية العليا اليسرى للصورة) وكما يأتي: **انظر المقطع البرمجي في الملحق.**

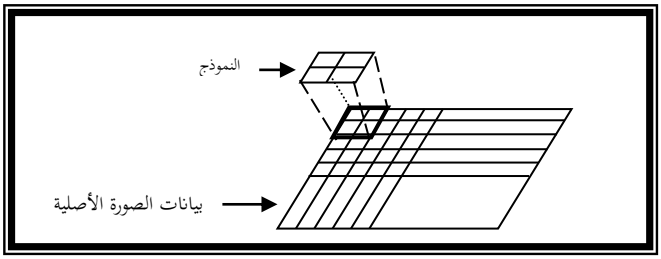
1. وضع النموذج على الصورة بحيث تتطابق مواقع الوحدات الصورية للنموذج على مواقع الوحدات الصورة للصورة الأصلية كما في الشكل(3).

2. نُطَبق نماذج الفحص جميعها المبينة في الجدول (2) على الصورة (كل نموذج منها على حدة) إلى أن نحصل على حالة مطابقة لأحد النماذج مع الصورة الأصلية، عندها نتوقف.

3. وضع الناتج في ملف ثانٍ.الناتج كما ذكرنا سابقا عبارة عن رمز (رمز النموذج المطابقة بياناته مع بيانات الصورة الأصلية) يتكون من جزأين يدل على النموذج المستخدم، إذ يمثل الجزء الأول من الرمز مجموع قيم معاملات (الواحدات) النموذج، والجزء الثاني يدل على التسلسل التصاعدي لبيانات النموذج ذات المجموع المتساوي، أما إذا كان مجموع الواحدات في النموذج = 4 أو 0 فسيتكون الرمز من جزء واحد فقط وهو المجموع فقط. حسب الجدول(2).

4. نزحف النموذج بمقدار وحدتين صورية إلى اليمين(أي عمودين إلى اليمين). الشكل(4–أ).

5. تعاد الخطوة (1-4) حتى نهاية الصف.



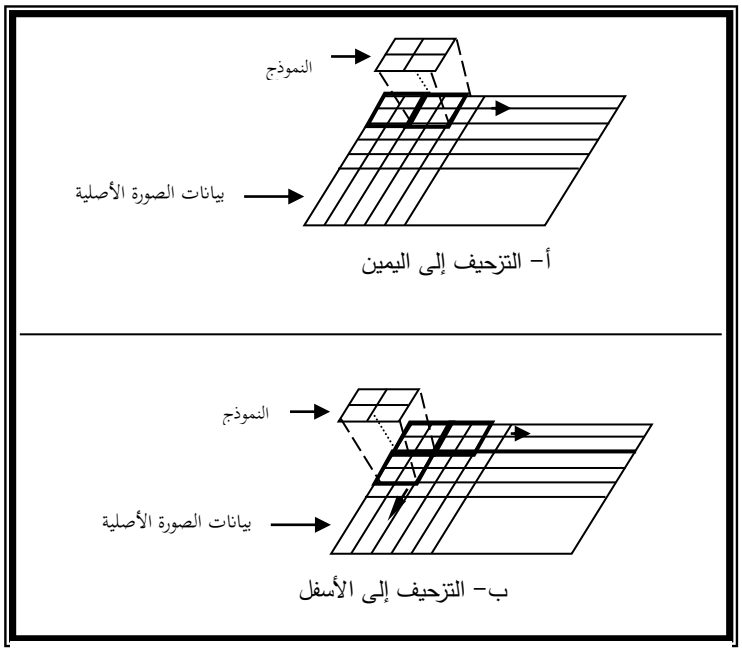
الشكل(3) رسم تخطيطي يوضح عملية التطابق

6. نزحف النموذج بمقدار وحدتين صورية إلى الأسفل(أي عمودين إلى الأسفل).الشكل(4–ب).

7. تعاد الخطوة (1-6) حتى نهاية الصورة.

وبذلك يتكون ملف يضم بيانات تتراوح قيمها ما بين (0 – 6) فقط كما يأتي:

```
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000222425000000000000000000000000000011000000000
0000000000111100001100000000000000000001121140022140000002212025000000
0223300011330000211200000000000000331200122202121000003312250000000222 2
0003231011331326434001200023120114120330132522222121003132003225000012
1225222200026334334434140003300042501131334344433344260002134113114334
4434344333422000000000000000310001125022000011121200000025000000000000
0000000000000134250021340022000013131400000025000000000000000000000000
0000000000022000000000000002500000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000001112121200000120000000000211112120000000000
0000000000000031131314250001134000000000000121225000000000000000000000
0000232201125250000400001125012003102525250000000000000000000004421
2125322121212131003221212522310223121322525000000000000000000000000013
2626013262626261400222626013330131426131414000000000000000000000000000
0000000011213404011340000000000000000000000000000000000000000000261400
1326000000000000000000000000000 0000000000000000000000000000000000000000
```



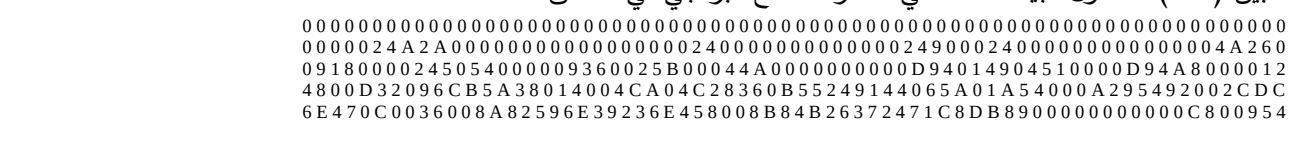
الشكل (4) رسم تخطيطي يوضح عملية التزحيف

المرحلة الثالثة:

يتم تحويل القيم الناتجة إلى أعداد ثنائية (1،0) حسب نظام الأعداد الثنائية، حيث تُمَثل كل قيمة بـ (3 bit) أي يُمثل 0 بالقيمة 000 و1 يُمثل بالقيمة 001 و 2 يُمثل بالقيمة 010 ... وهكذا. في هذه المرحلة يزداد حجم الملف الذي يؤدي إلى تعقيد الطريقة بزيادة البيانات، فضلا عن زيادة العشوائية بحيث توهم محطم الشفرة (المتطفل)، عليه سيزداد الوقت اللازم لتخمين الخوارزمية في حالة المهاجمة مما يزيد من كفاءة الطريقة. والناتج يصبح كما يأتي: انظر المقطع البرمجي في الملحق.

المرحلة الرابعة:

تُحول الأعداد الثنائية إلى أحد أنظمة الأعداد، مثلاً ليكن نظام الأعداد السداسي عشر (HEX)، إذ يتم اخذ كل 4 bits لتمثل قيمة لرقم واحد بذلك تصبح بيانات الملف تتراوح قيمتها مابين (0-F). فتتحول البيانات كا لاتي: انظر المقطع البرمجي في الملحق.



2 4 0 0 0 4 9 4 5 0 0 0 0 0 A 8 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 5 C 5 4 0 4 5 C 0 1 2 0 0 0 2 C B 3 0 0 0 0 0 5 4 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 9 0 0 0 0 0 0 0 0 0 0 0 0 5 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 4 9 4 5 1 4
0 0 0 0 A 0 0 0 0 0 0 0 0 1 1 2 4 A 2 8 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 3 2 5 9 6 6 2 A 0 0 2 5 C 0 0 0 0 0 0 0 0 0 0 2 8 A 5 4 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 1 3 4 8 1 2 A A A 0 0 1 0 0 0 0 9 5 4 1 4 0 3 2 1 5 5 5 5 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 9 1 1 4 5 5 6 9 1 4
5 1 4 5 9 0 1 A 4 5 1 5 5 2 6 4 2 4 C A 2 D 2 A A 8 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 D 6 5 8 1 6 B 2 C B 2 C 6 0 0 9 2 C B 0 2
D B 0 5 9 8 B 1 6 6 1 8 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 5 1 7 0 4 0 4 B 8 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 5 8 C 0 0 B 5 8 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

<u>المرحلة الخامسة:</u>

من الخصائص المشتركة لمعظم الصور ان النقاط المتجاورة تكون مترابطة بشكل كبير ولهذا السبب تحتوي على معلومات متكررة والفعالية المهمة في ذلك هي إيجاد تمثيل جديد لتلك النقاط باسلوب يقلل من التكرار[4]. لذا تم في هذه المرحلة استخدام احدى طرائق كبس الصور وهي RLE (Run Length Encoding) على بيانات الملف النهائي التي قيمتها مساوية للصفر فقط، (باعتبار ان بيانات الملف هي مصفوفة ثنائية وان عملية المسح ستتم صفاً بعد صف) إذ يتم حساب عدد الاصفار المتتابعة وتكوين رمز يتكون من جزأين، الاول 0 والثاني يمثل عدد الاصفار وذلك لانه لوحظ ان هذه القيمة تكون متكررة اكثر من غيرها. ثم تخزن البيانات النهائية مع حجم الصورة وحجم النماذج في ملف ذي امتداد (DAT.) كما بالشكل الأتي:

28 64 2 2 0 4B 2 4 A 2 A 0 11 2 4 0 D 2 4 9 0 3 2 4 0 E 4 A 2 6 0 2 9 1 8 0 4 2 4 5 0 1 5 4 0 5 9 3 6 0 2 2 5 B 0 3 4
4 A 0 A D 9 4 0 1 1 4 9 0 1 4 5 1 0 4 D 9 4 A 8 0 4 1 2 4 8 0 2 D 3 2 0 1 9 6 C B 5 A 3 8 0 1 1 4 0 2 4 C A 0 1 4 C
2 8 3 6 0 1 B 5 5 2 4 9 1 4 4 0 1 6 5 A 0 1 1 A 5 4 0 3 A 2 9 5 4 9 2 0 2 2 C D C 6 E 4 7 0 1 C 0 2 3 6 0 2 8 A 8 2 5
9 6 E 3 9 2 3 6 E 4 5 8 0 2 8 B 8 4 B 2 6 3 7 2 4 7 1 C 8 D B 8 9 0 B C 8 0 2 9 5 4 2 4 0 3 4 9 4 5 0 5 A 8 0 13 5 C
5 4 0 1 4 5 C 0 1 1 2 0 3 2 C B 3 0 5 5 4 0 1A 9 0 A 5 4 0 68 4 9 4 5 1 4 0 4 A 0 7 1 1 2 4 A 2 8 0 12 3 2 5 9 6 6 2
A 0 2 2 5 C 0 9 2 8 A 5 4 0 11 1 3 4 8 1 2 A A A 0 2 1 0 4 9 5 4 1 4 0 1 3 2 1 5 5 5 5 0 12 9 1 1 4 5 5 6 9 1 4 5 1 4
5 9 0 1 1 A 4 5 1 5 5 2 6 4 2 4 C A 2 D 2 A A 8 0 12 2 D 6 5 8 1 6 B 2 C B 2 C 6 0 2 9 2 C B 0 1 2 D B 0 1 5 9 8 B
1 6 6 1 8 0 1A 2 5 1 7 0 1 4 0 1 4 B 8 0 20 5 8 C 0 2 B 5 8 0 2F

إذ يمثل أول رقم (28) عدد الأسطر (Height) في الصورة الأصلية، أما الرقم الثاني (64) فيمثل عدد الأعمدة(Width) في الصورة، أما الرقم الثالث(2) والرابع (2) فيدل على حجم النموذج المستخدم، ثم يأتي ذلك بيانات النص، [في حالة أن قيمة البيانات 0 فان القيمة التي تليها هي عدد الأصفار المتتابعة (4B). وجميع البيانات الناتجة من هذه المرحلة تكون ممثلة بنظام الأعداد السداسي عشر.

خوارزمية فك الشفرة الطريقة المقترحة:

يعد كل من حجم النموذج وتسلسل معاملات النموذج وطريقة تكوين رموز البيانات مفتاحاً لتحليل الشفرة بهذه الطريقة، لذا على محلل الشفرة(المستلم) ان يعلم المفتاح إذ ان حجم النموذج يرسل ضمن الرسالة أما تسلسل معاملات النموذج فيتم الاتفاق عليه، بينما طريقة تكوين رموز البيانات تكون لدى الطرفين حسب خوارزمية معينة. والخطوات الأتية تمثل طريقة فك الشفرة:

1. إقرأ أول قيمة من بيانات الرسالة المشفرة وضعها في متغير اسمه (ROW) والذي يمثل عدد الصفوف في الصورة الحقيقية.

2. إقرأ القيمة التالية من بيانات الرسالة المشفرة وضعها في متغير اسمه (COL) الذي يمثل عدد الأعمدة في الصورة الحقيقية.

3. إقرأ حجم نماذج الفحص التي تمثل القيمتين الثالثة والرابعة من الرسالة المشفرة.

4. إعمل على تكوين جدول نماذج الفحص ورموز البيانات كما بينا سابقاً (تسلسل معاملات النموذج متفق عليها مسبقا).

5. حول كل قيمة من البيانات المتبقية إلى أعداد ممثلة بنظام الأعداد الثنائية وذلك بأخذ كل 4 bits لتمثل قيمة لعدد واحد (أي كل قيمة تمثل بـ 4 bits).

6. حول البيانات الناتجة في الخطوة (5) إلى أعداد عشرية تتراوح قيمتها بين(0-6)، إذ تؤخذ كل (3bits) لتمثل عدداً واحداً، ووضع الناتج في مصفوفة بحجم (ROW*COL).

7. أعطِ قيماً ابتدائية لـ (X=0 , Y=0).

8. اقرأ قيمة (N) من البيانات الناتجة في الخطوة (6).

9. افحص قيمة N :

- إذا كانت N=0 عندئذ نكتب البيانات كما في النموذج (أ) من الجدول(2) أي بالشكل($\begin{matrix}0&0\\0&0\end{matrix}$) في المصفوفة.
- أما إذا كانت N=4 فستتم كتابة البيانات كما في النموذج (ب) من الجدول (2) أي بالشكل ($\begin{matrix}1&1\\1&1\end{matrix}$) في المصفوفة.
- وإذا كانت N=1 عندئذ تتم قراءة قيمة أخرى من البيانات في الخطوة (6) ولتكن (NN) ثم يتم فحصها كالأتي:

إذا كانت NN=1 ، أي أن الرمز هو (11) فتكتب البيانات كما في النموذج (م) من الجدول (2) في مصفوفة النتيجة. أما إذا كانت NN=2 ، أي أن الرمز هو (12) فتكتب البيانات كما في النموذج (ن) من الجدول (2) في مصفوفة النتيجة، ...وهكذا بقية القيم.

- إذا كانت N=2 عندئذ تتم قراءة قيمة أخرى من البيانات في الخطوة (6) ولتكن (NN) ثم يتم فحصها كالأتي:

إذا كانت NN=1 ، أي أن الرمز هو (21) فتكتب البيانات كما في النموذج (ز) من الجدول (2) في مصفوفة النتيجة. أما إذا كانت NN =2 ، أي ان الرمز هو (22) فتكتب البيانات كما في النموذج (ح) من الجدول (2) في مصفوفة النتيجة، ... وهكذا بقية القيم.

وهكذا بنفس الطريقة أيضا تُثبت بيانات النماذج في مصفوفة النتيجة عندما تكون قيمة N =3.

10. تزداد قيمة كل من Y بمقدار 2.

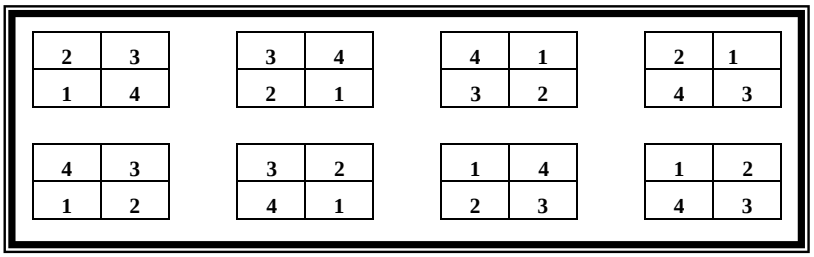
11. تستمر الخطوات (8-10) حتى نهاية الصف (Y=COL).

12. تزداد قيمة كل من X بمقدار 2.

13. تستمر الخطوات (8-10) حتى نهاية البيانات الناتجة من الخطوة (6) أي (X=ROW).

مناقشة سرية الطريقة وكفاءتها:

لتصميم نظام صحيح للتشفير هنالك معايير لقياس كفاءة النظام وهي معايير شانون [9]، فالخوارزمية المقترحة قد تقدم سرية عالية. أما المفتاح فهو بسيط ولكن لا يستطيع المتطفل استنتاجه بسهولة لأننا نستطيع التحكم بحجم النماذج وطريقة إعطاء تسلسل الخلايا في كل نموذج(الاتجاه) من نماذج الفحص. انظر الشكل (5).



الشكل(5) يبين بعض الاتجاهات المختلفة لتثبيت معاملات النموذج

أما بالنسبة إلى بساطة عملية التشفير وفك الشفرة فتعتبر بسيطة لأننا نعتمد على جهاز الحاسوب في ذلك. أما عن نسبة الخطأ فهي تساوي صفراً لان الطريقة تعد من طرائق معالجة الصور دون الفقدان في المعلومات وعند حساب مقياس المصداقية من نوع Root-Mean-Square error كانت النتيجة (eRMS=0). أما بالنسبة إلى تمديد الرسالة(extension of the message) فقد تمت زيادة حجم البيانات بشكل بسيط من خلال مراحل التشفير، وقد حصلت زيادة بسيطة بالحجم. ولكن عند مقارنة الحجم النهائي للملف مع ملف البيانات في الصيغة الثنائية فكان هناك نقصان بالحجم.

أما حين مناقشة الطريقة طبقاً لفرضية أسوأ الاحتمالات [9]، فإذا حصل المتطفل على خوارزمية التشفير فسيكون اعتماد سرية الطريقة على المفتاح (الذي يعتمد على حجم النموذج وتسلسل خلاياه) وطريقة تكوين الجدول الخاص بالنماذج. ولكن اذا علم المتطفل خوارزمية بناء الجدول فستكمن سرية الطريقة في:

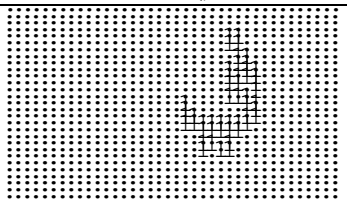
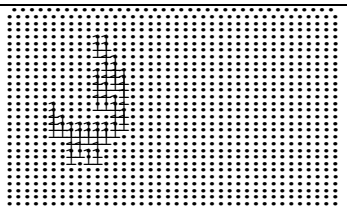
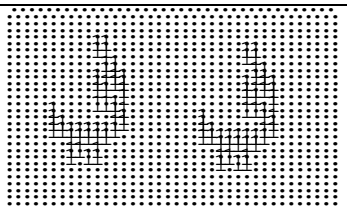
1. طريقة تمثيل رموز البيانات النهائية وخزنها لإرسالها عبر قنوات الاتصال إذ يضم الملف النهائي **المقدمة** (Header) مع الرسالة المرسلة الذي قد يوهم المهاجم بان المعلومات الموجودة في المقدمة تكون من ضمن الرسالة وبذلك يحصل على معلومات خطأ وعليه يحتاج إلى وقت أطول لفصل بيانات الرسالة الحقيقية عن المقدمة.

2. إن عدد(bits) المستخدمة لتمثيل الأعداد في المرحلة الثالثة ونظام الأعداد المستخدم لتحويل الأعداد الثنائية إلى أعداد أخرى في المرحلة الرابعة والذي يمكن ان يتغير باعتماد نظام اخر للاعداد وذلك بالاتفاق بين الطرفين مما يزيد من أمنية الطريقة.

3. حجم النماذج المستخدمة.

فان اصغر حجم للنموذج في هذه الطريقة هو 2*2 والذي يعطي 16 احتمالاً لقراءة البيانات من النموذج و(X) من الاحتمالات لطريقة تكوين رمز البيانات، (Z) من الاحتمالات لتمثيل البيانات في المرحلة الثالثة و(Y) من الاحتمالات لتمثيل البيانات في المرحلة الرابعة عليه سيكون أمام المتطفل (X*Y*Z*16) احتمال مبدئي ليطبقها جميعاً، وان عدد النماذج في كل جدول ($16=2^4$). أما إذا كان حجم النموذج 3*3 والذي يعطي 2^9 احتمالاً لقراءة البيانات من النموذج (أي تحديد الاتجاه) لتكوين رمز البيانات، عليه سنحتاج (X*Y*Z*512) احتمال، وعدد النماذج في كل جدول ($512=2^9$).. وإذا استُخدِم حجم النموذج 4*4 فعدد الاحتمالات سيكون كبيراً جداً ...وهكذا. اذاً سيكون لديه عدد كبير من الاحتمالات في حالة عدم علمه بالحجم الحقيقي، لذا بإمكاننا استخدام نماذج اكبر من 2*2 لجعل عملية معرفة المفتاح صعبة جداً.

أما احتمال أن يكون لدى المتطفل جزء من النص المشفر أو أن يكون لديه جزء من النص الصريح وما يقابله من النص المشفر فلن يمكنه من اكتشاف بقية الرسالة ولا يستطيع تخمين الخوارزمية و لا معرفة المفتاح لان أي رمز (أو كلمة) في الرسالة لا تتكرر بياناته المشفرة لو تكرر نفس الرمز (أو الكلمة) بموقع آخر، انظر المثال الأتي الذي يضم كلمة "في":

الصورة في الصيغة الثنائية	بيانات الملف النهائية
	1e 26 2 2 0 33 1 0 E 2 4 A 0 D B 4 0 C 2 3 6 0 C 1 1 4 C 0 A 4 8 0 2 D 8 0 A 9 3 4 8 A 2 E 0 C B 5 9 6 0 C 4 9 4 5 4 0 30
	1e 26 2 2 0 2E 9 2 A 0 D 2 2 0 D 9 6 C A 0 C 4 9 C 5 4 0 C 9 4 4 C 0 A 1 5 0 2 B 5 4 0 A 8 8 A 2 C 9 8 0 B 5 9 6 3 0 C 1 C 6 4 0 36
	1e 26 2 2 0 2E 9 2 A 0 5 4 4 0 7 4 4 0 5 B 0 7 2 5 B 2 8 0 4 4 9 5 0 6 4 9 C 5 4 0 4 6 A 0 6 1 2 8 9 8 0 4 2 C 9 0 5 A 8 0 1 5 A A 0 2 2 4 0 1 5 9 A 0 4 2 2 2 8 B 2 6 0 2 1 2 2 8 0 1 6 4 0 5 B 2 C 6 0 3 1 6 D C 8 6 0 5 1 C 6 4 0 4 2 4 A 2 8 0 C B 2 C C 0 22

الجدول(3) : مثال لبيانات رسالة تضم نفس الكلمة بمواقع مختلفة في الصورة.

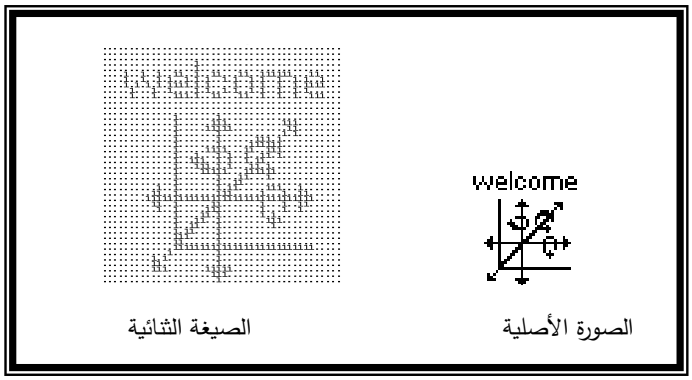
الاستنتاجات:

بما أن خوارزميات التشفير ممكن أن تكسر قبل وصول الرسالة إلى الطرف المعني وذلك عند توافر الوقت والمعلومات الكافية ليتمكن المتطفل من تحديد الخوارزمية المستخدمة في عملية التشفير أو إيجاد الضعف العام في الخوارزمية. لذا سعينا في هذا البحث لإيجاد طريقة تطيل من الزمن اللازم لمهاجمة الرسالة من قبل المتطفل إذ تم بناء خوارزمية باستخدام لغة C++ بحيث تحقق التوافق بين كل معايير شانون مجتمعة معا بخلاف الأنظمة السابقة وان الطريقة المقترحة في هذا البحث تكلف المهاجم وقتاً أطول وهي تتمتع بالصفات الأتية :

1. إن النص سوف يُؤخذ على شكل صورة (BMP) للاستفادة من طرائق معالجة الصور في التشفير، بذلك لا يستطيع المتطفل ان يستفيد من ترددات الحروف أي أننا بهذه الطريقة قد أضعنا على المتطفل فرصة الاستفادة من العمليات الإحصائية أو الحسابية فالبيانات ستكون على شكل

ثنائي [1,0]، بخلاف الطرائق المستخدمة سابقاً التي قد تعتمد على إيجاد مفكوك رقم ،أو تعتمد على الترددات الإحصائية للأحرف أو قد تستخدم LFSR أو NLFSR أو DES في التشفير الكتلي.

2. يمكن تشفير الخرائط والنصوص المكتوبة يدوياً أو المكتوبة بأسلوب صوري (الكتابة المسمارية مثلاً) او المطبوعة وبأية لغة.انظر الشكل (6،7،8).



الشكل(6)

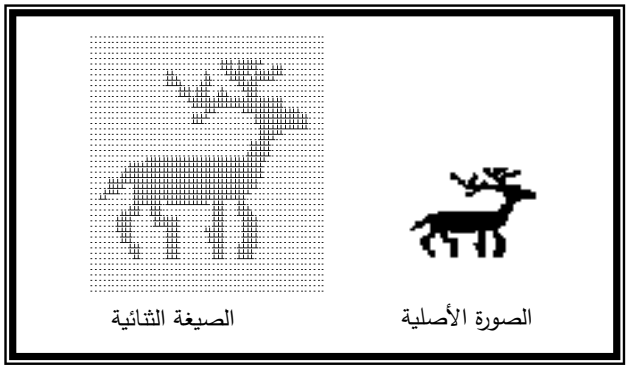
الملف النهائي للشكل(6):

```
2C 2C 2 2 0 16 2 4 0 B 5 1 2 0 1 A 2 4 A 4 8 2 2 0 1 9 2 8 9 4 5 1 2 8 2 2 0 3 A A 6 5 2 A D 3 2
9 2 4 0 1 C 5 5 2 4 8 2 A 9 2 4 8 A A 0 2 4 8 2 4 1 4 4 5 2 2 D 1 3 1 4 4 D 2 0 1 A A 4 5 A 2 5
0 16 2 4 0 2 4 8 0 C 4 8 0 1 2 6 3 4 0 3 5 C 0 7 1 2 0 2 2 4 0 2 4 4 A 4 C B 0 7 1 2 0 2 9 6 4 A
0 1 C A D 1 5 0 8 2 4 1 9 2 9 2 4 8 2 A C 9 8 6 0 8 1 2 0 1 5 C 6 5 3 0 1 C B 4 D 0 9 2 4 0 2 4
8 3 2 6 0 1 3 0 8 2 5 5 4 8 0 2 9 3 2 6 0 1 2 5 6 5 0 1 3 4 0 5 6 D C 6 D 6 5 9 6 8 B 2 C B 3 6
B 3 6 B 4 3 0 6 6 2 4 0 1 3 2 D 8 0 1 1 2 2 4 B 0 1 6 0 7 4 8 3 2 6 2 4 0 3 D 9 8 0 8 4 9 9 3 0
1 2 4 0 C 1 2 6 9 1 4 5 9 4 5 1 4 5 1 4 5 1 4 5 1 2 8 0 4 2 4 9 3 0 3 9 0 B 1 2 6 8 0 2 4 B 2 8
8 0 E 3 6 6 0 9
```

الشكل(7)

الملف النهائي للشكل(7):

```
34 2E 2 2 0 22 2 5 9 8 B 2 C B 2 C E 4 9 2 4 9 2 2 A 0 1 2 2 8 8 0 1 9 4 5 8
0 2 4 9 8 5 A C B 2 C B 2 C B 2 C 0 2 9 0 2 4 A 0 2 2 4 4 5 1 4 5 1 4 5 1 4
4 A 0 3 9 0 2 4 A 0 2 2 4 9 2 4 9 2 3 4 5 0 2 2 4 0 1 1 2 8 0 2 9 1 6 5 9 6
5 9 6 5 9 6 6 C A 0 2 4 8 0 1 2 5 0 2 1 2 2 2 8 A 2 8 A 2 8 C 9 0 2 1 2 0 2
9 4 0 2 4 8 B 2 C B 2 C B 0 5 2 4 0 1 1 2 8 0 2 6 E 4 9 2 4 9 2 4 9 2 4 8 0
1 2 5 0 2 2 5 9 9 2 4 9 2 4 7 1 6 5 9 6 5 9 6 5 8 0 1 4 A 0 2 9 4 9 2 4 9 2
4 9 2 2 A 0 7 B 5 9 6 5 9 6 5 A 4 8 B 2 C B 2 C 6 0 C 4 8 0 F 1 2 8 0 4 1 2
8 9 4 0 4 9 6 6 4 9 2 4 9 2 4 5 4 0 1 4 A 2 A 0 4 B 5 9 6 5 9 6 5 9 6 6 E 2
C B 3 6 E 1 8 1 2 8 A 8 0 6 1 2 4 9 1 5 0 3 4 A 2 A 0 2 9 1 5 0 1 1 2 8 E 2
C B 2 D 2 3 4 0 2 4 A 2 A 0 1 1 3 1 6 3 0 2 6 6 1 8 0 3 5 B 6 D 2 2 1 2 8 A
8 0 1 9 3 8 0 2 1 2 0 5 2 4 0 1 9 4 5 4 0 1 2 C C 0 2 9 4 5 9 4 5 1 4 5 1 4
5 1 4 5 1 6 5 A 4 5 9 8 A 8 0 4 9 4 9 2 4 9 2 4 9 2 4 9 1 5 0 5 1 6 0 1 2 C
B 2 C B 2 C B 2 C D C 7 1 6 5 8 C 0 D 5 A C 6 0 16
```



الشكل(8)

الملف النهائي للشكل(8):

```
30 38 2 2 0 35 4 A 2 4 A 2 8 9 4 0 C 8 8 0 2 9 0 1 5 C 6 6 3 6 8 B 3 0 B 8 D 0 2 C A A 0 1 B 6 D
A 8 B 2 C 0 C 9 4 4 A 0 1 6 4 A 0 1 D C 0 E 1 7 2 4 9 2 4 9 2 4 0 D 8 B 2 B 0 2 1 6 6 E 4 9 2 2
2 0 A 2 C 6 0 4 2 E 4 9 2 4 0 10 2 4 5 8 0 11 2 4 5 4 0 9 9 4 5 1 4 5 1 4 5 1 4 5 1 4 5 1 4 5 9
9 0 1 C 0 7 1 9 9 2 4 9 2 4 9 2 4 9 2 2 A 0 6 1 1 6 6 4 9 2 4 9 2 4 9 2 4 9 0 1 C 0 6 2 6 3 8 2
4 9 2 4 9 2 4 9 2 4 5 8 0 6 2 C 0 2 9 2 3 9 2 4 9 1 B 8 B 3 6 A 8 0 8 1 2 C C 1 1 5 0 4 9 4 0 1
9 4 2 8 0 7 1 2 0 2 9 4 8 A 8 0 1 1 2 8 0 1 4 8 0 8 6 E 0 3 4 5 4 0 1 2 E 4 0 1 2 4 0 8 1 6 D A
2 0 2 8 A 8 0 1 1 B 8 0 1 9 9 0 A B 8 0 2 8 A 8 0 1 1 2 8 1 2 8 A 8 0 46
```

3. نستطيع تغيير معادلة التشفير(عملية مطابقة نماذج الفحص مع بيانات الصورة) باستخدام علاقات منطقية (NOT,NAND,OR,AND,XOR,…) .

4. إذا أراد شخص تغيير مقطع في الرسالة يتطلب ذلك إعادة تشفير الرسالة كاملةً.

5. إن استخدام إحدى طرائق كبس الصور مثل RLE لكبس جزء من بيانات الملف النهائي سيزيد من تعقيد الطريقة. يمكننا تغيير الطريقة المستخدمة بالكبس اعتمادا على البيانات.

6. لكي تُكسَر الشفرة يجب على محطم الشفرة معرفة نوع الرسالة وعليه معرفة حجم نماذج الفحص المستخدمة واتجاه تسلسل الخلايا فيها، وطريقة تمثيل البيانات بعد كل مرحلة وكيفية خزن البيانات النهائية.جميع هذه النقاط مجتمعةً يجب أن تكون متوافرة لدى المتطفل لكي يستطيع فقط ان يخمن

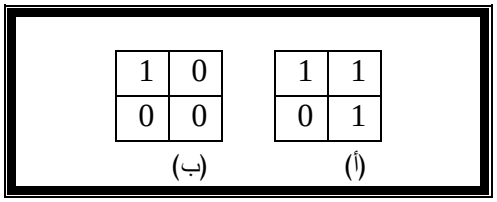
جميع احتمالات النماذج لكي يتمكن من فك الشفرة أو تحطيمها، وهذا بحد ذاته يحتاج إلى وقت. ان فقدان أية معلومة من هذه المعلومات يجعل عملية تخمين الخوارزمية وتوقعها عملية صعبة. لذا يمكن ان تعد هذه الطريقة من الطرائق الجيدة عملياً.

7. لا نحتاج إلى إرسال نماذج الفحص عبر قنوات الاتصال.

قد يخطر في ذهن القارئ إن عملية تحويل بيانات الصورة الثنائية الى (0-6) باستخدام نماذج الفحص ثم التحويل إلى الرموز الثنائية يمكن أن يُختَصر وذلك بتحويل بيانات الصورة الى الرموز الثنائية مباشرة بدون تطبيق النماذج، أي عندما ينطبق احد النماذج ستتم كتابة رمز النموذج مباشرة.

مثال: إذا كان النموذج المبين بالشكل (6–أ) ينطبق على بيانات الصورة، عندئذ نخزن الرمز (1101)، وعندما ينطبق النموذج في الشكل (6–ب) نخزن الرمز (1000). وذلك للتقليل من الحجم. عندئذ نحن نقول ان الهدف من البحث هو إخفاء معالم الرسالة سواء أكانت نصية أم مخططات أم خرائط لمواقع، وان الهدف من التشفير بهذه الطريقة هو: اولاً: زيادة العشوائية في البيانات الناتجة بحيث يصعب على المتطفل تخمين الطريقة. ثانياً: لكي نُطيل من الزمن اللازم في عملية كسر الشفرة.

لذا فان مراحل التشفير بهذه الطريقة تزيد من تعقيد الطريقة إذ تغير من معالم الرسالة بعد كل مرحلة، وعليه تزداد العشوائية ونحصل على سرية اكبر.



الشكل(9) يبين عينة من نماذج الفحص.

المصادر

(1) التشفير، (2002) ، الموسوعة العربية للكمبيوتر والانترنيت
http://www.c4arab.com/showac.php2a=:d103 from[6]

(2) التشفير، (2002)، مفاهيم الانترنيت
http://www.itep.co.ae/itportal/arabic/content/educationalcenter/internetconcept/encryption.asp.(a) from[6]

(3) الحمداني، وسيم عبد الامير ، (1992)، "انظمة التشفير الانسيابي"، الجامعة التكنلوجية– بغداد

(4) النعيمي، ميسون خضر حسين،(2003)، " كبس صور الوثائق النصية العربية "، بحث ماجستير ،كلية علوم الحاسبات والرياضيات،جامعة الموصل، الموصل.

(5) بروست، بوزورث، ترجمة الدكتور المهندس ميثم محمود، (1989)، " الرموز والشفرات والحاسبات، مقدمة الى امن المعلومات"، الجامعة التكنلوجية– بغداد– الطبعة الاولى.

(6) جلميران، ميلاد جادر سعيد، (2003)، "تشفير النصوص العربية باستخدام شفرة مورس"، بحث ماجستير ،كلية علوم الحاسبات والرياضيات،جامعة الموصل، الموصل.

(7) كونزالز، رافائيل؛ وينتز، يول ، (1989)، معالجة الصور الرقمية.

[8] AL-Etewi, Raya Jassim Esaa, (2001), “Design Hybrid Cryptography System and Attacking Streem Cipher Using Neural Network”, M.Sc. Research, College of Computer Sciences and Mathematics, University of Mosul, Iraq.

[9] Beker, Henry and Piper, Fred (1982) **Cipher Systems the Protection of Communications**, PP. 162-166.

[10] Farshad, Zargham, (2000) “Home Connections need Better Security”, Electronic Engineering Time, Issue 142. ,PP. 98. (from [8])

[11] Lamberieux, P. (1999) “Encryption”, NORMAN-Norman Data Defense System.

[12] Wiley, John and Sons (1996) “Applied Cryptography Protocols, Algorithms and Source Code in C”, Second Edition, Inc.

الملحق

المقطع البرمجي الخاص بالمرحلة الثانية

```
FILE oout,txt;
int **data_thin;    /* مصفوفة البيانات الثنائية */
int m0[2][2]  = {{0,0},{0,0}};  int m4[2][2] = {{1,1},{1,1}};
int m11[2][2] = {{1,0},{0,0}};  int m12[2][2]= {{0,1},{0,0}};
int m13[2][2] = {{0,0},{1,0}};  int m14[2][2]= {{0,0},{0,1}};
int m21[2][2] = {{1,1},{0,0}};  int m22[2][2]= {{0,1},{0,1}};
int m23[2][2] = {{0,0},{1,1}};  int m24[2][2]= {{1,0},{1,0}};
int m25[2][2] = {{0,1},{1,0}};  int m26[2][2]= {{1,0},{0,1}};
int m31[2][2] = {{0,1},{1,1}};  int m32[2][2]= {{1,0},{1,1}};
int m33[2][2] = {{1,1},{0,1}};  int m34[2][2]= {{1,1},{1,0}};
/*-------------------------------------------------------*/
```

نماذج الفحص

اجراء ملء المصفوفة بالبيانات الثنائية

```
void fill_matrix_fron_binary_file_out(int **data,int row,int
col)
 {
      int n=999;
      y = 0;
      while ( ( !feof(txt) ) && (y <row) )
      {
      x=0;
      while ( (!feof(txt) ) &&  ( x < col) )
       {
         fscanf(txt ,"%i", &n) ;
         data[y][x]=n;
         x++;
       }
       y++;
     }
 }
/*-------------------------------------------------------*/
```

اجراء مطابقة النماذج

```
void templates_matching(int **data,char *s,int nr,int nc)
  {
  int j,kk,y_c,sw,mm,p1,p2,p3,p4;

   if ((oout = fopen("out2.d", "w"))== NULL) /* output file*/
         {
          fprintf(stderr, "Cannot open input file.\n");
          getch();
          exit(0);
          }
   flush(oout);
   fprintf(oout,"%i %i \n",nr,nc);
   int x_c = 0;
   int nx=nr-2;
   int ny=nc-2;
   while (x_c <= nx )
```

```
   {
     y_c = 0;
     while (y_c <= ny)
       {
      flag=0;
     int sm =                                  /*مجموع الواحدات في النموذج*/

data[x_c][y_c]+data[x_c+1][y_c]+data[x_c][y_c+1]+data[x_c+1][
y_c+1];

     switch (sm)
     {
      case 0:flag=1; fprintf(oout,"%i ",0);     break;
                                                          /* الرمز =0 */

      case 1:int x = 1 ;
                                                  /* مجموع الواحدات =1 */
      while ((flag==0)&&(x <=4 ))
      {
      switch (x)
      {case 1 :p1= data[x_c][y_c]&&m11[0][0];
      p2=!(data[x_c][y_c+1]||m11[0][1]);                  مطابقة النموذج
p3=!(data[x_c+1][y_c]||m11[1][0]);
p4=!(data[x_c+1][y_c+1]||m11[1][1]);
                           sw=(p1&&p2&&p3&&p4);
                           if (sw==1)  {flag=1;
                                        fprintf(oout,"%i
",1);
                                        fprintf(oout,"%i
",4);}
                                                          /* الرمز =14 */
                            else   x++;
                           break;
                  case 2 :p1=!(data[x_c][y_c]||m12[0][0]);
                           p2= data[x_c][y_c+1]&&m12[0][1];
                           p3=!(data[x_c+1][y_c]||m12[1][0]);

p4=!(data[x_c+1][y_c+1]||m12[1][1]);
                           sw=(p1&&p2&&p3&&p4);
                           if (sw==1)   {flag=1;
                                         fprintf(oout,"%i
",1);
                                         fprintf(oout,"%i
",3);}
                                                  /* الرمز =13 */
                            else x++;
                           break;
                  case 3 :p1=!(data[x_c][y_c]||m13[0][0]);
```

```
                              p2=!(data[x_c][y_c+1]||m13[0][1]);
                              p3=data[x_c+1][y_c]&&m13[1][0];

p4=!(data[x_c+1][y_c+1]||m13[1][1]);
                              sw=(p1&&p2&&p3&&p4);
                              if (sw==1) {flag=1;
                                          fprintf(oout,"%i ",1);
                                          fprintf(oout,"%i ",2);
}
                                                          /* الرمز =12 */
                               else x++;
                              break;
                     case 4 :p1=!(data[x_c][y_c]||m14[0][0]);
                              p2=!(data[x_c][y_c+1]||m14[0][1]);
                              p3=!(data[x_c+1][y_c]||m14[1][0]);
                              p4=data[x_c+1][y_c+1]&&m14[1][1];
                              sw=(p1&&p2&&p3&&p4);
                              if (sw==1) {flag=1;
                                          fprintf(oout,"%i ",1);
                                          fprintf(oout,"%i ",1);
}
                                                          /* الرمز =11 */
                               else x++;
                              break;
                        }
                    }
                 break;
       case 2:x = 1 ;
                                                          /* مجموع الواحدات =2 */
              while ((flag==0)&&(x <=6 ))
                { switch (x)
                    {
                    case 1 :p1=data[x_c][y_c]&&m21[0][0];
                              p2=data[x_c][y_c+1]&&m21[0][1];
                              p3=!(data[x_c+1][y_c]||m21[1][0]);

p4=!(data[x_c+1][y_c+1]||m21[1][1]);
                              sw=(p1&&p2&&p3&&p4);
                              if (sw==1) { flag=1;
                                           fprintf(oout,"%i
",2);
                                           fprintf(oout,"%i
",6); }
                                                          /* الرمز =26 */
                               else    x++;
                              break;
                     case 2 :p1=!(data[x_c][y_c]||m22[0][0]);
                              p2=data[x_c][y_c+1]&&m22[0][1];
```

```
                              p3=!(data[x_c+1][y_c]||m22[1][0]);
                              p4=data[x_c+1][y_c+1]&&m22[1][1];
                              sw=(p1&&p2&&p3&&p4);
                              if (sw==1) {flag=1;
                                          fprintf(oout,"%i ",2);
                                          fprintf(oout,"%i
",2);}
                                                          /* الرمز =22 */
                              else   x++;
                              break;
                     case 3 :p1=!(data[x_c][y_c]||m23[0][0]);
                              p2=!(data[x_c][y_c+1]||m23[0][1]);
                              p3=data[x_c+1][y_c]&&m23[1][0];
                              p4=data[x_c+1][y_c+1]&&m23[1][1];
                              sw=(p1&&p2&&p3&&p4);
                              if (sw==1) { flag=1;
                                           fprintf(oout,"%i
",2);
                                           fprintf(oout,"%i
",1);}
                                                          /* الرمز =21 */
                               else  x++;
                              break;
                     case 4 :p1=data[x_c][y_c]&&m24[0][0];
                              p2=!(data[x_c][y_c+1]||m24[0][1]);
                              p3=data[x_c+1][y_c]&&m24[1][0];

p4=!(data[x_c+1][y_c+1]||m24[1][1]);
                              sw=(p1&&p2&&p3&&p4);
                              if (sw==1) { flag=1;

                                           fprintf(oout,"%i
",2);
                                           fprintf(oout,"%i
",5);}
                                                          /* الرمز =25 */
                               else x++;
                              break;
                     case 5 :p1=!(data[x_c][y_c]||m25[0][0]);
                              p2=data[x_c][y_c+1]&&m25[0][1];
                              p3=data[x_c+1][y_c]&&m25[1][0];

p4=!(data[x_c+1][y_c+1]||m25[1][1]);
                              sw=(p1&&p2&&p3&&p4);
                              if (sw==1) { flag=1;
                                           fprintf(oout,"%i
",2);
                                           fprintf(oout,"%i
",3); }
```

```
                                                    /* الرمز =23 */
                                else x++;
                               break;
                        case 6:p1=data[x_c][y_c]&&m26[0][0];
                               p2=!(data[x_c][y_c+1]||m26[0][1]);
                               p3=!(data[x_c+1][y_c]||m26[1][0]);
                               p4=data[x_c+1][y_c+1]&&m26[1][1];
                               sw=(p1&&p2&&p3&&p4);
                               if (sw==1) { flag=1;
                                            fprintf(oout,"%i
",2);
                                            fprintf(oout,"%i
",4);}
                                                    /* الرمز =24 */
                                else x++;
                               break;
                           }
                      }
                    break;
   case 3: x = 1 ;
                                                    /* مجموع الواحدات =3 */
              while ((flag==0)&&(x <=4 ))
                {  switch (x)
                    {
                    case 1 :p1=!(data[x_c][y_c]||m31[0][0]);
                               p2=data[x_c][y_c+1]&&m31[0][1];
                               p3=data[x_c+1][y_c]&&m31[1][0];
                               p4=data[x_c+1][y_c+1]&&m31[1][1];
                               sw=(p1&&p2&&p3&&p4);
                               if (sw==1) { flag=1;
                                            fprintf(oout,"%i
",3);
                                            fprintf(oout,"%i
",1);}
                                                    /* الرمز =31 */
                                else x++;
                               break;
                        case 2 :p1=data[x_c][y_c]&&m32[0][0];
                               p2=!(data[x_c][y_c+1]||m32[0][1]);
                               p3=data[x_c+1][y_c]&&m32[1][0];
                               p4=data[x_c+1][y_c+1]&&m32[1][1];
                               sw=(p1&&p2&&p3&&p4);
                               if (sw==1) {flag=1;
                                           fprintf(oout,"%i ",3);
                                           fprintf(oout,"%i
",2);}
                                                    /* الرمز =32 */
                                else   x++;
```

```
                        break;
                case 3 :p1=data[x_c][y_c]&&m33[0][0];
                        p2=data[x_c][y_c+1]&&m33[0][1];
                        p3=!(data[x_c+1][y_c]||m33[1][0]);
                        p4=data[x_c+1][y_c+1]&&m33[1][1];
                        sw=(p1&&p2&&p3&&p4);
                        if (sw==1) {flag=1;

                                    fprintf(oout,"%i ",3);
                                    fprintf(oout,"%i
",3);}
                                                        /* الرمز = 33 */
                         else   x++;
                        break;
                case 4 :p1=data[x_c][y_c]&&m34[0][0];
                        p2=data[x_c][y_c+1]&&m34[0][1];
                        p3=data[x_c+1][y_c]&&m34[1][0];

p4=!(data[x_c+1][y_c+1]||m34[1][1]);
                        sw=(p1&&p2&&p3&&p4);
                        if (sw==1){ flag=1;
                                    fprintf(oout,"%i ",3);
                                    fprintf(oout,"%i
",4);}
                                                        /* الرمز = 34 */
                         else x++;
                        break;
                    }
                }
               break;
      case 4:flag=1;
                                                /* مجموع الواحدات = 4 */
            fprintf(oout,"%i ",4);
                                                        /* الرمز = 4 */
            break;
      }
 y_c++; y_c++;
{
x_c++;  x_c++;
{fclose(oout);}
/*-------------------------------------------------*/
void stage2(int **data_thin,int row,int col)
}
  fill_matrix_fron_binary_file_out(data_thin,row,col);
  templates_matching(data_thin,"out2.d",row,col);
 }
/*-------------------------------------------------*/
```

```
void main_cod()
 {
 if ((txt = fopen("bmp-b.d", "r"))== NULL) /* input file */
         {
          fprintf(stderr, "Cannot open input file.\n");
          getch();
          exit(0);
          }
  flush(txt); rewind(txt);
  int row,col;
  fscanf(txt,"%i%i", &row,&col);
  stage2(data_thin,row,col);
  fclose(txt);
  delete(data_thin);
{
```

/*---------- المقطع البرمجي الخاص بالمرحلة الثالثة ----------
----*/

"تحويل كل رقم الى 3 Bit"

```
void stage3()
 {int  n = 999 ,
  if ((oout = fopen("out-b.d", "w"))== NULL) /* output file*/
         {
          fprintf(stderr, "Cannot open input file.\n");
          getch();
          exit(0);
    }
  flush(oout);
  while (!feof(txt) )
  {
   fscanf(txt ,"%i", &n) ;
   switch (n)
   {
     case 0 :fprintf(oout,"%i %i %i ",0,0,0);    break;
     case 1 :fprintf(oout,"%i %i %i ",0,0,1);    break;
     case 2 :fprintf(oout,"%i %i %i ",0,1,0);    break;
     case 3 :fprintf(oout,"%i %i %i ",0,1,1);    break;
     case 4 :fprintf(oout,"%i %i %i ",1,0,0);    break;
     case 5 :fprintf(oout,"%i %i %i ",1,0,1);    break;
     case 6 :fprintf(oout,"%i %i %i ",1,1,0);    break;
   }
  }
 fclose(oout);
 }
/*-------------------------------------------------*/
void main_cod()
 {
 if ((txt = fopen("out2.d", "r"))== NULL) /*input file*/
         {
```

```
          fprintf(stderr, "Cannot open input file.\n");
          getch();
          exit(0);
          }
  flush(txt);  rewind(txt);
  int row,col;
  fscanf(txt,"%i%i", &row,&col);
  stage3();
  fclose(txt);
 }
```

/*---------- المقطع البرمجي الخاص بالمرحلة الرابعة ----------*/

"التحويل الى HEX "

```
FILE binary_txt;    /* input file */
/*-----------------------------------------------------*/
void stage4()
 {
int  n1 = 999 ,n2 = 999 ,n3 = 999 ,n4 = 999; /* initial values*/
  if ((oout = fopen("hex.d", "w"))== NULL)  /* output file */
         {
          fprintf(stderr, "Cannot open input file.\n");
          getch();
          exit(0);
          }
   flush(oout);
   while (!feof(txt) )
   {
   fscanf(txt ,"%i", &n1) ;
   if (!feof(txt) )  fscanf(txt ,"%i", &n2) ;
       else n2=0;
   if (!feof(txt) )  fscanf(txt ,"%i", &n3) ;
       else n3=0;
   if (!feof(txt) )  fscanf(txt ,"%i", &n4) ;
       else n4=0;
   int sm=n1+n2+n3+n4;
   switch (sm)
   {
case 0 : fprintf(oout,"%i ",0);  break;      /*0000*/
case 1 :if (n4==1) fprintf(oout,"%i ",1);   /*0001*/
      else if (n3==1)   fprintf(oout,"%i ",2);   /*0010*/
      else if(n2==1) fprintf(oout,"%i ",4);     /*0100*/
      else if(n1==1)fprintf(oout,"%i ",8);      /*1000*/
            break;
case 2 :if( (n3==1)&&(n4==1)) fprintf(oout,"%i ",3) ; /* 0011 */
      if ((n2==1)&&(n4==1))  fprintf(oout,"%i ",5) ;  /* 0101 */
      if ((n2==1)&&(n3==1))  fprintf(oout,"%i ",6) ;  /* 0110 */
      if ((n1==1)&&(n4==1))  fprintf(oout,"%i ",9) ;  /* 1001 */
```

```
      if ((n1==1)&&(n3==1))  fprintf(oout,"%c ",'A') ; /* 1010 */
      if ((n1==1)&&(n2==1))  fprintf(oout,"%c ",'C'); /* 1100 */
            break;
case 3 :if (n4==0) fprintf(oout,"%c ",'E');         /*1110*/
        else if (n3==0) fprintf(oout,"%c ",'D');    /*1101*/
        else  if (n2==0) fprintf(oout,"%c ",'B'); /*1011*/
        else if(n1==0) fprintf(oout,"%i ",7);    /*0111*/
             break;
case 4 : fprintf(oout,"%c ",'F');         /*1111*/
             break;
      }
    }
 fclose(oout);
{

/*-----------------------------------------------------*/

void stage_four()
 {
if ((binary_txt = fopen("out-b.d", "r"))== NULL)/* input file */
         {
          fprintf(stderr, "Cannot open input file.\n");
          getch();          exit(0);
          }
 flush(binary_txt);  rewind(binary_txt);
 int row,col;
 fscanf(binary_txt,"%i%i", &row,&col);
 stage4();
 fclose(binary_txt);
 }
```