On Subliminal Cryptography

***Sh. S. Al-Bundi**

Date of acceptance 9/11/2005

Abstract

In recent years, cryptography has played a big role especially in computer science for information security. Generally, there are several approaches in cryptography such as classical methods, stream cipher, block cipher and public-key methods. The common feature among them is the cipher text is unreadable or not understood. But in the new approach (Subliminal approach), the cipher text is readable or understood, this is an important feature. In this paper we will present two algorithms for subliminal cryptography with examples.

Introduction

Cryptography and information security are considered one of important sciences in the world, especially after using the computers in these sciences. On other hand, the information and technology revolution and the armament race between the big powers give these sciences more priority because these sciences play a big role in the espionage field, information thievery and policy forecasting. In general, after information revolution, several trends have appeared in the cryptography [1, 2], these are: -

- Classical methods.
- Stream cipher methods.
- Block cipher methods.
- Public-Key methods.

In the sections below we will illustrates the concepts of each method: -

Classical Methods

It has three types: -

1. Substitution Methods

In this method, we will reorder the positions of characters in the plain text (using several methods) to produce the ciphered text.

2. Monoalphabetic Method

In this method, we will shift the characters of the plain text depending on a key to produce the ciphered text.

3. Polyalphabetic Method

In this method, we will propagate the key word on the plain text to produce the ciphered text.

Stream Cipher Methods

These methods depends on binary number system; polynomial characteristics and linear and non-linear shift register with using initial value. The figure (1) illustrates the block diagram of stream cipher method [1].

The main properties of stream cipher are:

- Implementation at high speed.
- High complexity especially with non-linear shift registers.
- Easy implementation.
- Using with important applications such as speech cryptography.
- Symmetric type.

There are several algorithms that depend on this method such as GEFFE, HADMARD, PLHSS, A5, RAMBUTAN, GIFFORD and so on.

* Dep. of Math, College of Education / Ibn Al-Haitham, Baghdad University

Block Cipher Methods

They are similar to the previous methods but they treat the information as blocks of bit instead of one by one bit. They have static algorithms. One of important types is DES (date Encryption Standard), which has 16 rounds with 64 bits as a block length. This method is a symmetric method. There are several algorithms, which depend on this method such as LUCIFER, MADRYGA, NEWDES, FEAL, REDOC, LOKI, RC2, IDEA, KHUFU & KHAFRE, MMB, GOST, CAST, SAFER, RC5 and so on [2].

Public-Key Methods

They are important methods because the key of decryption process is not the encryption process (also its big prime numbers). Therefore, these methods are a symmetric type. There are several algorithms that depend on this method such as (Knapsack) and (RSA) [1]. The following steps illustrate the RSA algorithm:

1. Let X& Y big prime numbers.
2. Z = X*Y.
3. Select E a prime number where GCD (E, Φ(Z)) – 1.
4. D is equal to inverse (E, D) mod Φ(Z).
5. Cipher = Message ^{E} mod Z.
6. Message = Cipher ^{D} mod Z.

Concept of text Generation

Computer text generation is the process of constructing text (phrases, sentences, paragraphs) in a natural language in the sense that it is the opposite of natural language understanding by machine. Although this problem has been investigated for 25 years, few coherent principles have emerged, and the approaches have varied widely [3]. Attempts at generating text have been made with two general research goals:
(a) generating random sentences to test a grammar or grammatical theory and (b) converting information from an internal representation into a natural language [4].

Random Generation

This approach, the random generation of text constrained by the rules of a test grammar, is of limited interest to workers in Artificial Intelligence, since it is oriented more toward theoretical linguistics than toward functional natural-language-processing systems. The objective of implementing a generation system of this sort is to test the descriptive a adequacy of the test grammar as illustrated by the following two systems.

Victor Yngve (1962) [3] was one of the first researchers to attempt English text generation; the work was seen as preliminary to a full program for machine translation. Yngve used a generative context-free grammar and a random number generator to produce grammatical sentences. The system randomly selected one production from among those that were applicable at each point in the generation process, starting from those productions that produced <Sentence> and finally randomly selecting words to fill in the <Noun>, <Verb>, and other like positions. This is an example of text produced by the system:

The water under the wheels in oiled whistles polished shiny big and big trains is black.

Joyce Friedmans (1969, 1971) [3] system was designed to test the effectiveness of transformational grammars. It operated by generating phrase markers (derivation tree) and by performing transformations of them until a surface structure was generated. The generation was random, but the user could specify an input phrase marker and semantic restrictions between various terminals in order to test specific rules for grammatical validity.

These two systems, while relevant to work in linguistic, are only peripherally related to recent work in artificial intelligence. The fundamental emphasis in AI test generation work has been on the meaning, as opposed to the syntactic form of language.

Discourse In Language Generation

The majority of natural language researches deal with the structure of the separated sentences. In terms of considering the theoretical discussion of these issues, some of these researches consider the connectivity of sentences and coherence of discourse, as in English language [5, 6]. It is worthy mentioning that this issue has not been taken, to provide Arabic references. As a main topic relating with Arabic computational linguistics, considering the language generation as an output from data base systems. In spoken or written language the focusing phenomena occur at many of discourse. In a discourse, a particular element that speaker usually centers his attention on is called discourse element [3, 4].

Proposed Subliminal Methods

The real deception of enemy is the unfelt deception. Depending on this concept our encryption approach is suggested. It converts the plain text to encrypted plain text that means the enemy cannot feel any encryption operation.

As we have already said, the idea of our method is to produce the encrypted plain text from plain text via algorithms that depend on natural languages processing especially text generation field. By this method the enemy will not feel by encrypted text (at least till the message expire is dead). Therefore, this method differs from the recent methods in idea. For example, we will say, "we will hit at night", we will send "we will play with Japan team".

Now we display the algorithms we implement it as follows:

The First Algorithm

Encryption

```
Counter = 7
Cipher text = plain text + Location the letter in plain text – Counter
Counter = Counter – 1
Until no plain text
```

Decryption

```
Counter = 7
Plaintext = Cipher text – Location (Order) the Word in the cipher text + Counter
Counter = Counter – 1
Until no cipher text
```

Example: -

If we have plain text [I'm Happy], then we cipher it as follows:

I, Code (I) = 8, Location in plain text = 1, Counter = 7

Cipher = Code (I) + Location (I) – Counter = 8 + 1 – 7 = 2.

The letter which is represented by the code (2) is C, and then we can represent C in the word Curry.

M, Code (M) = 12, Location in plain text = 2, Counter = 6

Cipher = Code (M) +Location (M) – Counter = 12 +2 – 6 =8.

The letter which is represented by the code (8) is I, and then we can represent I in the word Ibsen

H, Code (H) = 7, Location in plain text = 3, Counter = 5

Cipher = Code (H) +Location (H) – Counter = 7 +3 – 5 =5.

The letter which is represented by the code (5) is F, and then we can represent Fin the word Found

A, Code (A) = 0, Location in plain text = 4, Counter = 4

Cipher = Code (A) +Location (A) – Counter = 0 +4 – 4= 0.

The letter which is represented by the code (0) is A, and then we can represent A in the word Adams.

P, Code (P) = 15, Location in plain text = 5, Counter = 3

Cipher = Code (P) +Location (P) – Counter = 15+5 – 3= 17.

The letter which is represented by the code (17) is R, and then we can represent R in the word Rate.

P, Code (P) = 15, Location in plain text = 6, Counter = 2
Cipher = Code (P) +Location (P) – Counter = 15 + 6 – 2 = 19.
The letter which is represented by the code (19) is T, and then we can represent Tin the word The.
Y, Code (Y) = 24, Location in plain text = 7, Counter = 1
Cipher = Code (Y) +Location (Y) – Counter = 24 + 7 – 1 = 30 mod $_{26}$ = 4.
The letter which is represented by the code (4) is E, and then we can represent E in the word Euphonium.
∴The cipher text:

Curry Ibsen found Adam's rate the euphonium.

To decryption the message we will do the following:

Curry, Code (C) = 2, Location the word in the cipher text= 1, Counter = 7

Plain = Code (C) – Location the word + Counter =2– 1 +7 = 8

The letter which is represented by the code (8) is I, and it's location in the message is 1.
Ibsen, Code (I) = 8, Location the word in the cipher text= 2, Counter = 6

Plain = Code (I) – Location the word + Counter =8–2 +6 = 12

The letter which is represented by the code (12) is M, and it's location in the message is 2.

Found, Code (F) = 5, Location the word in the cipher text= 3, Counter = 5

Plain = Code (F) – Location the word + Counter =5– 3 +5 = 7

The letter which is represented by the code (7) is H, and it's location in the message is 3.

Adams, Code (A) = 0, Location the word in the cipher text= 4, Counter = 4

Plain = Code (A) – Location the word + Counter =0– 4 +4 = 0

The letter which is represented by the code (0) is A, and it's location in the message is 4.

Rate, Code (R) = 17, Location the word in the cipher text= 5, Counter = 3

Plain = Code (R) – Location the word +Counter=17–5 +3 =15

The letter which is represented by the code (15) is P, and it's location in the message is 5.

The, Code (T) = 19, Location the word in the cipher text= 6, Counter = 2

Plain = Code (T) – Location the word + Counter =19 – 6 +2= 15

The letter which is represented by the code (15) is P, and it's location in the message is 6.

Euphonium, Code (E) = 4, Location the word in the cipher text = 7, Counter = 1

Plain = Code (E) – Location the word + Counter = 4 – 7 + 1 = 2$_{\text{mod } 26}$ = 24

The letter which is represented by the code (24) is Y, and it's location in the message is 7.

∴The plain text: I'm Happy

The Second Algorithm

When we want to cipher a text by using this algorithm, we will take each word in this text, firstly compute the number of characters in this word, then we generate a word that must start with character which has a code equal to that number. After that, we apply the following equation to each character in this word:

Ciphertext = (Plaintext + Length Plaintext) $_{\text{mod } 26}$.

Example:

If we want to cipher the plain text:

I'm Happy, then we do the following:

1. Take the next word in this text = I'm
2. Length of word = 2
3. Generate a word which must star with character which has code = 2,
   code (2) = C, generating word = Clip.
4. Implement the following equation for all the characters in this word:

Ciphertext = (Plaintext + length Plaintext)$_{\text{mod } 26}$

Cipher = (plain (I) + length Plain)$_{\text{mod } 26}$ = $(9 + 2)_{\text{mod } 26} = 11 \Rightarrow$ L, generating word = Light.

Cipher = (plain (M) + length Plain)$_{\text{mod } 26}$ = $(13 + 2)_{\text{mod } 26} = 15 \Rightarrow$ P, generating word = Panel.

Now we repeat steps 1, 2, 3, 4, for all words in this text.

Take the new text word = Happy, length = 5, Code (5) = F, generating word = Faris, then we apply the equation:

Cipher = (plain (H) + length Plain)$_{\text{mod } 26}$ = $(8 + 5)_{\text{mod } 26} = 13 \Rightarrow$ N, generating word = Not.

Cipher = (plain (A) + length Plain)$_{\text{mod } 26}$ = $(0 + 5)_{\text{mod } 26} = 5 \Rightarrow$ F, generating word = Found.

Cipher = (plain (P) + length Plain)$_{\text{mod } 26}$ = $(16 + 5)_{\text{mod } 26} = 21 \Rightarrow$ V, generating word = Volvo.

Cipher = (plain (P) + length Plain)$_{\text{mod } 26}$ = $(16 + 5)_{\text{mod } 26} = 21 \Rightarrow$ V, generating word = Volkswagen.

Cipher = (plain (Y) + length Plain)$_{\text{mod } 26}$ = $(24 + 5)_{\text{mod } 26} = 3 \Rightarrow$ D, generating word = Dodge.

The cipher text = Clip Light Panel. Faris Not Found Volvo, Volkswagen, Dodge

If we want to decipher a text which is ciphered by using the second algorithm we following the steps below:

1. Take the first word and separate the first character from it, then take its code which represents the number of characters in the original (word).
2. We count word from the text as the number of the code, then we separate the first character from each word, and apply the following equation:
   Plain = (Cipher – First Char)$_{\text{mod } 26}$.
3. We still repeat the step 1 and 2 for all the words in the text.

If we want to decipher the cipher text = Clip Light Panel. Faris Not Found Volvo, Volkswagen, Dodge, we do the following:

1. Take the first word = Clip, code(c) = 2
2. We repeated the following process for the two next word:

Take the next word = Light, L = 11

Plain = $(11 - 2)_{\text{mod } 26} = 9$, code (9) = I.

Take the next word = Panel, P = 15

Plain = $(15 - 2)_{\text{mod } 26} = 13$, code (13) = M.

Take the new word = Faris, code (F) = 5, that means the number of characters in the original word = 5, for this we repeat the following process for the next five words form the text:

Take the next word = Not, code (N) = 13.

Plain = $(13 - 5)_{\text{mod } 26} = 8$, code (8) = H.

Take the next word = found, code (F) = 6.

Plain = $(5 - 5)_{\text{mod } 26} = 0$, code (0) = A.

Take the next word = Volvo, code (V) = 21.

Plain = $(21 - 5)_{\text{mod } 26} = 16$, code (16) = P.

Take the next word = Volkswagen, code (V) = 21.

Plain = $(21 - 5)_{\text{mod } 26} = 16$, code (16) = P.

Take the next word = Dodge, code (D) = 3.

Plain = $(3 - 5)_{\text{mod } 26} = 24$, code (24) = Y.

The plain text = I'm Happy.

Implementation of Subliminal Methods

When we want to implement subliminal methods, we must have:

1. Natural language dictionary.
2. Natural language grammars with their semantics.
3. Natural language story generation techniques.

The figure (2) illustrates the block diagram of subliminal methods.

In spite of the fact that subliminal methods are good ciphering methods for deceiving the enemy, I believe that the perfect performance of subliminal method will be achieved when we use it as an intermediate ciphering between plain text and cipher text using the pervious mentioned methods. Figure (3) illustrates this idea.

Conclusion

To cipher the plain text we have several methods as we mentioned this in our introduction, some of these are good methods such as block cipher methods and public – key methods. But, the deceiving or the trick is not available because the cipher text is unreadable, therefore it is ciphered. Using the subliminal methods will increase the deception and trick via the ciphered plain text, this is the power point in subliminal methods. If we use subliminal methods as an intermediate method between the real plain text and ciphered text, we will obtain big trick. Why? Because if any one succeeds in break the cipher text, he will find the ciphered plain text, it is readable and understood therefore he will stop (at most). But if he mistrusts, he will try again with the ciphered plain text, which is considered another obstacle.

References

1. Baker H. &Piper F. 1982, " Cipher System: The Protection Communication", Northwood Publication, U. K.
2. Bruce S., 1996, "Applied Cryptography Protocols Algorithms and source Code in C ", John Wiley & Sons Inc., U.S.A.
3. Shapiro S., 1987, "Encyclopedia of Artificial Intelligence", U.S.A.
4. Mary D. H., 2000, "Introduction to Natural Language Processing", Layola University.
5. Patric, H. W., 1992, "Artificial Intelligence", Addison – Wesley Publishing, U.S.A.
6. Rich E. & Knight K., 1991, "Artificial Intelligence", McGraw – Hill Inc., U.S.A.

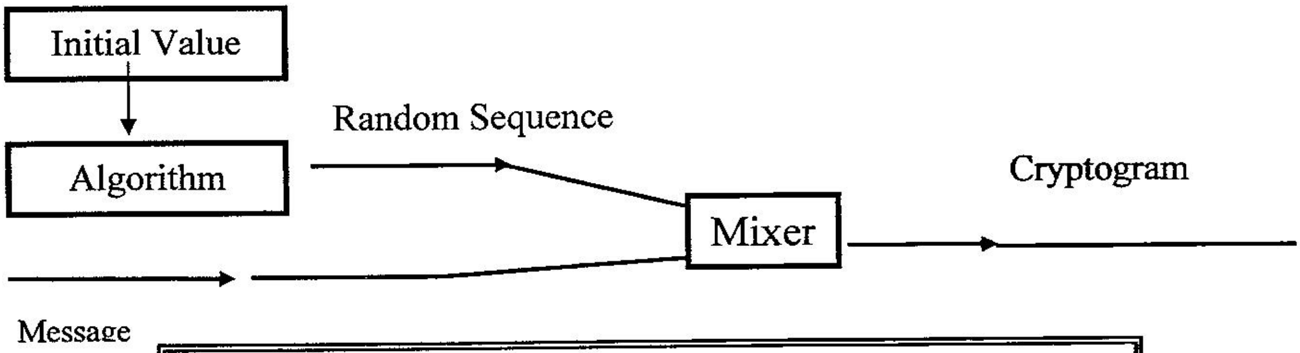

Figure 1: Block Diagram of Stream Cipher System

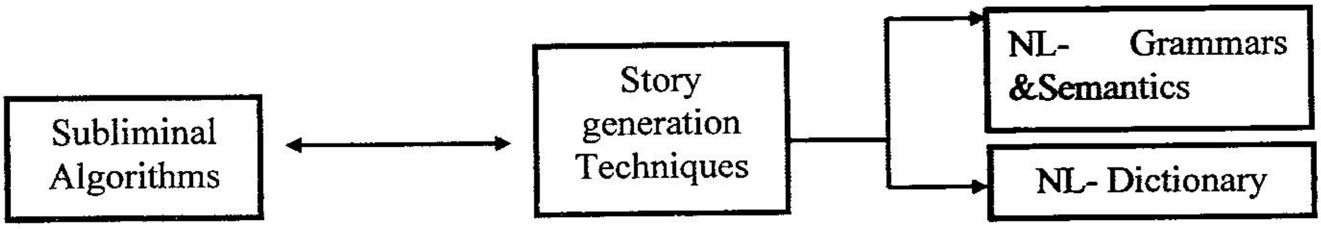

Figure 2: Subliminal Block Diagram

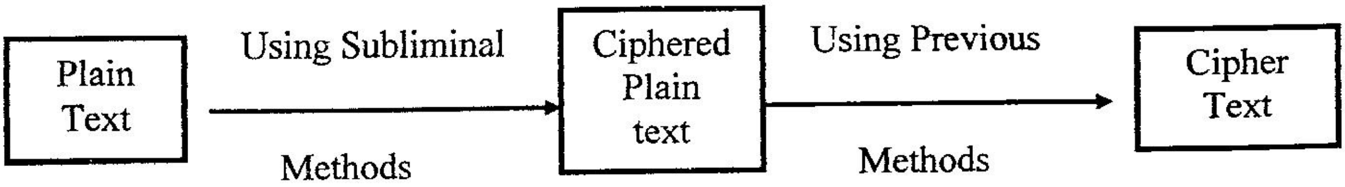

Figure 3: Prefect Performance of Subliminal Methods

تشفير القنوات غير المحسوسة

شيماء سلمان البندي *

* قسم الرياضيات ، كلية التربية / ابن الهيثم ، جامعة بغداد.

المستخلص

في السنوات الاخيرة لعب علم التشفير دوراً مهماً وخصوصاً في علم الحاسبات وأمن المعلومات. وعموماً توجد عدة طرق لتشفير المعلومات مثل الطرق التقليدية، طرق التشفير التسلسلي، طرق التشفير الكتلي وطرق تشفير المفتاح العام. والميزة المشتركة في هذه الطرق هي أن النص المشفر يكون غير مقروء أو غير مفهوم عند قراءته. ولكن في تشفير البيانات باستخدام طرق القنوات غير المحسوسة يكون النص المشفر نصاً مقروءاً ومفهوماً وهذه هي النقطة المهمة في هذا النوع من التشفير. في هذا البحث سنقدم طريقتان (خوارزميتان) لتشفير المعلومات بطريقة القنوات غير المحسوسة.