Using Genetic Algorithm to Break Knapsack Cipher with Sequence Size 16

Safaa S Omran [1], Ali S Al_Khalid [2], Israa F Ali [3]

[1] College of Elec. & Electronic Techniques / University of Baghdad
omran_safaa@ymail.com

[2] College of Elec. & Electronic Techniques / University of Baghdad
mudariben@yahoo.com

[3] Foundation of Technical Education
ms_israaf89@yahoo.com

Abstract: *With the growth of networked system and applications such as eCommerce, the demand for effective internet security is increasing. Cryptology is the science and study of systems for secret communication. It consists of two complementary fields of study: cryptography and cryptanalysis.The genetic algorithm is one of the search methods, which finds the optimal solution. It is one of the methods, which is used to decrypt cipher.This work focuses on using Genetic Algorithms to cryptanalyse knapsack cipher. The knapsack cipher is with a knapsack sequence of size 16 to encrypt two characters together. Different values of parameters have been used: Population size, mutation rate, number of generation.*

الخلاصة: مع نمو النظام والتطبيقات مثل التجارة الإلكترونية الشبكية، فإن الطلب على فعالية أمن الإنترنت آخذ في الازدياد.ال (Cryptologyraphy) هو علم ودراسة نظم الاتصالات السرية. وتتكون من حقلين مكملين: ال (Cryptology) و ال (Cryptanalysis).الخوارزمية الجينية هي واحدة من طرق البحث، التي تجد الحل الأمثل. وهي واحدة من الطرق التي تستخدم في فك الشفرة. يركز هذا العمل على استخدام الخوارزميات الجينية لفك شفرة نابساك. نابساك ذات ١٦ عنصرلتشفير حرفين معا. وقد استخدمت قيم مختلفة من المعايير: حجم السكان، ومعدل الطفرة، عدد الاجيال.

1. Introduction

With more and more developments in the field of computer networks and internet, the need for network, computer and information security is also increasing. There are different ways to secure information passed over the network. One such a technique is cryptology. Cryptology is the science and study of systems for secret communication [1]. Cryptography is the science of building new powerful and efficient encryption and decryption methods. It deals with the techniques for conveying information securely. The basic aim of cryptography is to allow the intended recipients of a message to receive the message properly while preventing eavesdroppers from understanding the message. Cryptanalysis is the science and study of method of breaking cryptographic techniques i.e. ciphers. In other words it can be described as the process of searching for flaws or oversights in the design of ciphers [2].

Among the useful ciphering systems is the knapsack ciphers [1]. One of the first knapsack ciphers was suggested by Merkle and Hellman in 1978[3]. It represented one of the initial attempts at a public key cryptosystem. While the cipher is based on an NP_complete problem [4]. This paper focuses on attack on knapsack cipher with a knapsack sequence of size 16 using Genetic Algorithm (GA). Genetic Algorithms are optimization and search techniques based on the principles of genetic and natural selection [5]. They contain three main operators: selection, crossover and mutation [6].

2. Knapsack Ciphers

Knapsack cipher was proposed by Merkle and Hellman in 1978 which is based on the NP-complete problem [2,7]. Given n objects each with a known volume and a knapsack of fixed volume, there is a subset of n objects which exactly fill the knapsack, another way to express the problem is given; say 7 numbers (3, 7, 12, 8, 22, 31, 16) is there some combination of these numbers which add to exactly 46? The list of numbers corresponds to the volume of the 7 objects while 46 is the fixed volume of the knapsack. In this case the answer is yes, 3+12+31=46. The only way to discover that answer, however, is by trial and error. Search all the possible combinations of the 7 numbers until one that produces the target sum is found. With 7 numbers there are only 128 possible combinations [4].

The knapsack cryptosystem belongs to major categories of public/private key cryptosystem [2]. The public/private key aspect of this approach lies in the fact that there are actually two different knapsack problems – referred to as the easy knapsack and hard knapsack. The Markle-Hellman algorithm is based on this property [2,7]. The easy knapsack is the private key. The hard knapsack is the public key [2].

The easy knapsacks have a sequence of numbers that are superincreasing. That is each number is greater than the sum of the previous numbers. Such a sequence is (3,5,9,18,38,75,155,312,628,1265,2536,5077,10157,20317,4

0639,81280)[4].To encrypt the plaintext message using easy knapsack are done by the following steps:

The plaintext block transforms into binary string $m = m_1,...,m_n$,where n is the length of block.

Selecting easy knapsack sequence $A' = \left(a'_1,..., a'_n \right)$,where n is the number of elements in knapsack sequence. The length of block is equal to the number of elements in knapsack sequence.

Computing target sum $(c1) = m_1 a_1 + m_2 a_2 + ... + m_n a_n$.

For example:

To encrypt the message kf do the following:

kf transforms into binary string , $\mathrm{m} = 0110101101100110.$ where ASCII of k is 01101011 and ASCII of f is $01100110.$

$$A' = \begin{pmatrix} 3, 5, 9, 18, 38, 75, 155, 312, 628, 1265, 2536, \\ 5077, 10157, 20317, 40639, 81280 \end{pmatrix}.$$

target sum $(c1) = 65276$.

The knapsack solution with the superincreasing sequence proceeds as follows. The target sum is compared with a greatest number in the sequence. If the target sum is smaller than this number, the knapsack will not fill, otherwise it will. Then the smaller element is subtracted from the target sum, and the result of the subtraction, is compared with next element. Such operation is done until the smallest number of sequence is reached. If the target sum is reduced to 0 value, a solution exists. In other case solution doesn't exist [2,7].

The superincreasing knapsack is easy to decipher, which means that it does not protect the message . Anyone can recover the bit pattern from the target sum for a superincreasing knapsack if the elements of the superincreasing knapsack are known. Merkle and Hellman suggested that such a simple knapsack be converted into a trapdoor knapsack which is not superincreasing and so is difficult to break.

3. Merkile-Hellman Knapsack Public-key Encryption

A. Converting Easy to Hard Knapsack

Each entity creates a public key and a corresponding private key [8].
An integer n is fixed as a common system parameter.
Alice should perform steps 3 – 7.

Choosing a superincreasing sequence $A' = \left(a'_1,..., a'_n \right)$.

$$2 * a'_n$$

Selecting a random integer $u >$

Selecting a random integer w, such that $\gcd(w, u) = 1$.

Finding inverse of $w \bmod u$

Computing $A_i = wA'_i \bmod u$ for $i = 1,2,..., n$ where $A = (a_1, a_2,..., a_n)$.

Alice's public key is A; Alice's private key is (u, w, A').

For the above example:
n=16.
Alice should perform steps 3 – 7.

$$A' = \begin{pmatrix} 3, 5, 9, 18, 38, 75, 155, 312, 628, 1265, 2536, \\ 5077, 10157, 20317, 40639, 81280 \end{pmatrix}.$$

$u > 2 * 81280, u = 162573$

$w = 13$.

w^{-1} is $100045 (100045 * 13 = 1 \bmod 162573)$

$$A = \begin{pmatrix} 39,65,117,234,494,975,2015,4056,8164,16445, \\ 32968,66001,132041,101548,40588,81202 \end{pmatrix}$$

B. Encryption and Decryption

Bob encrypts a message m for Alice, which Alice decrypts [8].

Encryption. Bob should do the following:

(a) Obtaining Alice's authentic public key $(a_1, a_2,..., a_n)$.

(b) Representing the message m as a binary string of length $n, m = m_1 m_2 ... m_n$.

(c) Computing the integer $c = m_1 a_1 + m_2 a_2 + ... + m_n a_n$.

(d) Sending the ciphertext c to Alice.

Decryption. To recover plaintext m from c, Alice should do the following:

(a) Computing $c1 = w^{-1} c \bmod u$.

(b) By solving a superincreasing subset sum problem, find the message bits are $m_i = m_1,..., m_n, m_i \in \{0,1\}$, such that $c1 = m_1 a_1 + m_2 a_2 + ... + m_n a_n$.

For the above example:

Bob encrypts a message $\mathrm{m} = \mathrm{kf}$ for Alice, which Alice decrypts.

Encryption Bob should do the following:

(a)

$$A = \begin{pmatrix} 39,65,117,234,494,975,2015,4056,8164,16445, \\ 32968,66001,132041,101548,40588,81202 \end{pmatrix}$$

(b) $\mathrm{m} = \mathrm{kf} = 0110101101100110.$

(c) $c = 198296.$

(d) Send the ciphertext c to Alice.

Decryption

To recover plaintext m from c, Alice should do the following:

(a)

$c1 = w^{-1} c \bmod u = 100045 * 198296 \bmod 162573 = 65276.$

.

(b) By solving a super increasing subset sum problem, the message bits are $\mathrm{m} = \mathrm{kf} = 0110101101100110.$

4. Genetic Algorithms

Genetic algorithms were developed by John Holland as a modification of what is called evolutionary programming [4].

Holland's idea was to construct a search algorithm modeled on the concepts of natural selection in the biological sciences. The result is a directed random search procedure. The process begins by constructing a random population of possible solutions. This population is used to create a new generation of possible solutions which is then used to create another generation of solutions, and so on. The best elements of the current generation are used to create the next generation. It is hoped that the new generation will contain "better" solutions than the previous generation [4,9].
The steps of genetic algorithm are as the following:

1- A random population of chromosomes is selected.
2- A fitness value for each chromosome in the population is determined.
3- The selection operation is made.
4- The crossover operation is made.
5- The mutation process is executed to produce new population.
6- Step 2 is repeated for the new population [2,7].

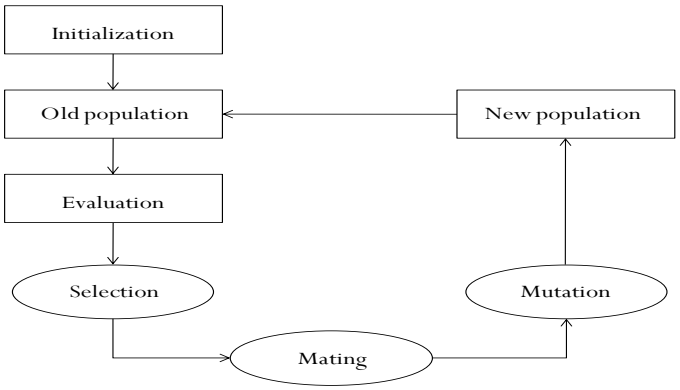

Figure 1: The basic genetic algorithm cycle.

Three processes which have a parallel in human genetics are used to make the transition from one population generation to the next. They are selection, mating and mutation. The basic genetic algorithm cycle based on these three processes is shown in Figure (1). Selection process determines which strings in the current generation will be used to create the next generation. The mating process determines the actual form of the strings in the next generation. At this point, two of selected parents are paired. The final step is one of mutation. A fixed small mutation probability is set at the start of the algorithm. Bits in all the new strings are then subject to change based on this mutation probability [4].

5. Cryptanalysis of knapsack Cipher Using Genetic Algorithm

Spillman [4] suggested genetic algorithm to solve the knapsack problem. Figure (2) shows the Markle-Hellman cryptosystem and cryptanalysis by means of genetic algorithm. The cryptanalysis starts from cipher text, which has an integer form. Each number represents a target sum of hard knapsack problem. The goal of the genetic algorithm is to translate each number into the correct knapsack, which represents the ASCII code for the plaintext characters.

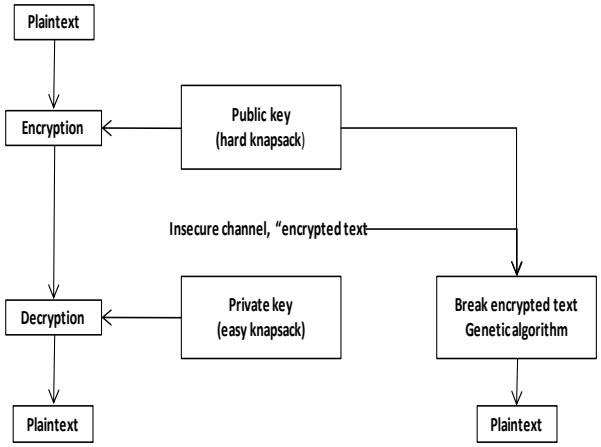

Figure 2: Markle-Hellman cryptosystem and cryptanalysis by means of Genetic Algorithm.

Encoding

The following restriction have been made for encoding
(1) Only the ASCII code will be encrypted.
(2) The super increasing sequence will have 16 elements; this number of elements guarantees that two characters have a

unique encoding.

Initialization

A random population of chromosomes (binary string 0's and 1's) is generated. The number of bits in each chromosome is equal to the number of elements key (i.e. 16).

Evaluation

In our work we used the following fitness function [1] to evaluate the generated individuals. Based on the fitness value obtained, it can be determined whether the optimal solution is reached or not.

$$fitness = \begin{cases} 1-\left(\frac{|Target - Sum|}{Target}\right)^{\wedge\frac{1}{2}} & if\ Sum \le Target \\ 1-\left(\frac{|Target - Sum|}{MaxDiff}\right)^{\wedge\frac{1}{6}} & if\ others \end{cases}$$

Where,
MaxDifference = max(Target, FullSum − Target)
Target is the ciphertext.
Sum is the sum of the current chromosome.
FullSum is the sum of all components in the knapsack.

Based on the fitness function given in the equation above the fitness value evaluates how the given sum is close to the target value for the knapsack. The value of the fitness function should be in the range of 0 to 1. Fitness value 1 indicates an exact match with the target sum for the knapsack. If the value of sum is greater than targets then it have a lower fitness value of chromosome, in this way it produces the infeasible solution. If the value of sum is less than target then it will produce a high fitness value and produce feasible solutions. Feasible solutions have a greater chance of being followed by the algorithm. Small differences between the current chromosome and the target sum should be amplified.

Selection

The important part of algorithm is selection of a new population. Selection of individuals is done according to their fitness value obtained. In our work we used the stochastic universal sampling selection. Stochastic universal sampling selection procedure may be implemented as follows:

1- The fitness function is evaluated for each individual, providing fitness values, which are then normalized. Normalization means dividing the fitness value of each individual by the sum of all fitness values, so that the sum of all resulting fitness values equals 1.
2- The population is sorted by descending fitness values.
3- Accumulated normalized fitness values are computed (the accumulated fitness value of an individual is the sum of its own fitness value plus the fitness values of all the previous individuals). The accumulated fitness of the last individual should be 1 (otherwise something went wrong in the normalization step).
4- A random number R between 0 and 1 is chosen.
5- The selected individual is the first one whose accumulated normalized value is greater than R.

Elite

Elite children are the individuals in the current generation with the best fitness values. These individuals automatically survive to the next generation [10].

Crossover

The single point crossover operation is applied in the algorithm. Single point crossover is shown in table (1).

Table (1) Single point crossover .

Chromosome1	10110101 | 00101111
Chromosome 2	10110000 | 11001100
Offspring 1	10110101 | 11001100
Offspring 2	10110000 | 00101111

Mutation

After crossover is performed, mutation takes place. Bit inversion is the type of mutation is used in this work. Bit inversion mutation process is shown in table (2).

Table (2) bit inversion mutation.

Parent	0010101100011101
Child	0111100100110100

6. Results

Genetic algorithms have been applied to cryptanalyses knapsack cipher successfully in short time. This paper used Genetic Algorithms to cryptanalyses knapsack cipher. The knapsack cipher is with a knapsack sequence of size 16.

The number of generations is 60, the population size is 1000, the selection type is stochastic universal sampling, the crossover type is single point crossover and this point is selected randomly, the crossover probability is 0.69, the mutation type is reversing, the mutation probability is 0.3, 6 bit from 16 bit are reversed in the mutation and elite is 0.01 are used in this paper.

The super increasing sequence used for the knapsack cryptosystem was (3,5,9,18,38,75,155,312,628,1265,2536,5077,10157,20317 ,40639,81280) and values of u, w are randomly selected at each run. The value of w^{-1} is computed according to the values of u, w. The hard knapsack is changed at each run according to the value u.

In our work the word "macro" is encrypted. The total time to obtain the word "macro" is 51.5337 second. The software (MATLAB 2009A) has been used. Pentium CORE^{TM}i5, processor 2.30GHZ, and RAM 4GB.

Figure (3) shows the best and mean fitness for each character in the word "macro". The best fitness of characters "ma" in the word "macro" becomes 1 in the generation 4 therefore the characters "ma" is obtained in the generation 25 as shown in Figure (3-a). While the characters "cr" in the word "macro" is obtained in the generation 46 shown in Figure (3-b) .The best fitness of character "o" in the word "macro" becomes 1 in the generation 12. Figure (4) shows the best fitness of all character in the word "macro". The character "o" in the word "macro" is obtained first, then the characters "ma", in the word "macro" then the characters "cr" in the word "macro".

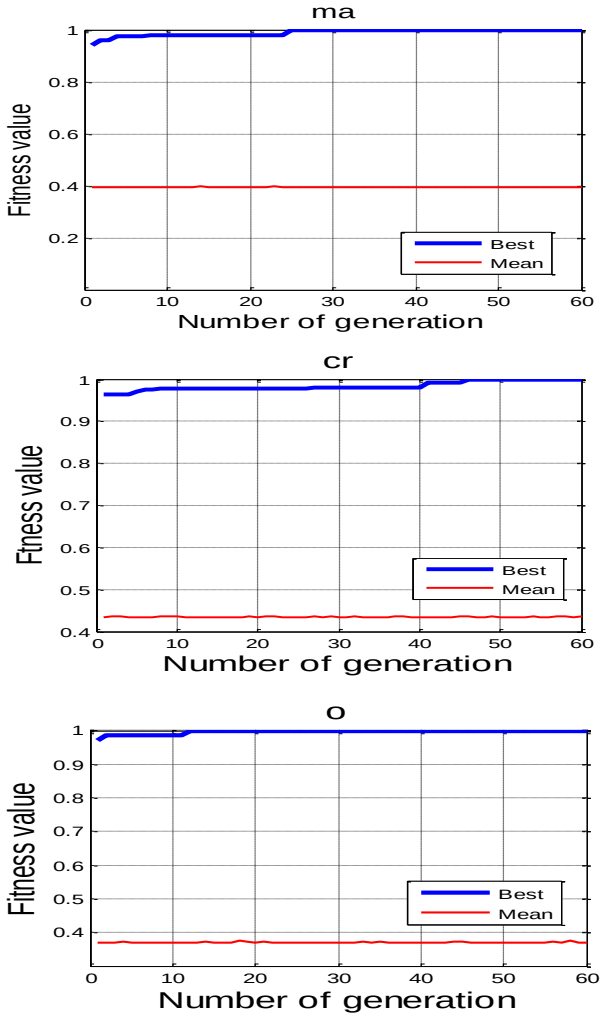

Figure 3: Best and mean fitness for each character in the word macro

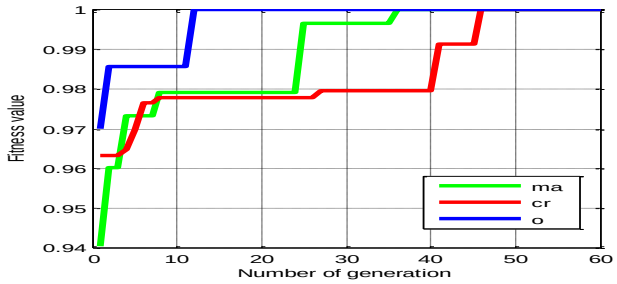

Figure 4: Best fitness versus Number of generation

Conclusions

This paper presents the attack of knapsack cipher of knapsack sequence of size 16 using Genetic Algorithm. This leads to the cryptanalysis of plaintext encrypted using knapsack cipher of knapsack sequence size 16 to encrypt two characters at the same time (8 bit ASCII code for each character).

This paper indicates that the efficiency of genetic algorithm attack on knapsack cipher can be improved by variation of mutation, crossover operation and size of population. The results are worse when the size of population decreases. The initial population size is inversely proportional to number of generations. The genetic algorithm offers a powerful tool for the cryptanalysis of knapsack cipher.

References

[1] R. Geetha, and L. Balasubramanian, " Genetic Algorithm solution for Cryptanalysis of Knapsack Cipher with Knapsack Sequence of Size 16 ,"International Journal of Computer Applications ,11(35):(0975 – 8887), December 2011.

[2] G. Poonam, and S. Aditya, "An Improved Cryptanalytic Attack on Knapsack Cipher using Genetic Algorithm," International Journal of Information Technology, 3(3):(145-15), 2006.

[3] R. C. Merkle and M. E. Hellman, " Hiding Information and Signatures in Trapdoor knapsacks,". IEEE Transactions on Information Theory.IT-24: 525-530, 1978.

[4] R. Spillman, "Cryptanalysis of knapsack ciphers using genetic algorithms," Cryptologia,17(4):367–377, October 1993.

[5] R. L. Haupt, and S. Ellen Haupt, " Practical genetic algorithms," 3rd addition, Wiley Interscience, 2004.

[6] A. P. Engelbrecht , " Computational Intelligence:An Introduction," 2nd Edition, University of Pretoria South Africa, 2007.

[7] G. Poonam, S. Aditya, and D.C. Agarwal, " An Enhanced Cryptanalytic Attack on

[8] Knapsack Cipher using Genetic Algorithm," Proceedings of World Academy of Science,

[9] Engineering and Technology, 12:(1307-688), March 2006.

[10] J. Alfred, "Handbook of Applied Cryptography," CRC press, 1996.

[11] R. Spillman , "Solving Large Knapsack Problems with a Genetic Algorithm," IEEE,1(95) :(632-637), 1995.

[12] MATLAB. 2009 "Genetic Algorithms and Direct Search ToolboxTM 2" User's Guide.