ISSN 1512-1232

PERSPECTIVES OF THE USE OF GENETIC ALGORITHMS IN CRYPTANALYSIS

Lali Beselia
Sokhumi State University, Politkovskaia str., Tbilisi, Georgia

Abstract

Modern cryptosystems analysis is a complex task, the solution of which is estimated to exceed the volume of cryptanalyst real opportunities. Evolutionary algorithms, namely genetic algorithms, have an opportunity to dramatically reduce the number of assumptions. Merkle - Hellman cryptosystem is a public key cipher, which is based on the famous knapsak task. Spilman was first,who used a genetic algorithm to break the cipher, then a scientist has expanded its work in this direction. In these works carried out known-plaintext attack. Yet there is the Shamir's polynomial-time algorithm, which the successfully breakable Merkle - Hellman system, finds the secret key via the public key. Despite the fact that the Shamir's algorithm is polynomial-time, for breaking a real system need to perform a large amount of computation.

The paper aims to demonstrate the advantages of the use of genetic algorithms for cryptoanalysis compared with other methods. Directly the public key cryptalgorithms cases, where we can finds the secret key via the public key. In this sense, we have created two separate and different from the other algorithms algorithm (the genetic libraries) have implemented this algorithm attack Merkle - Hellman cryptosystem. The obtained results allow us to conclude that the use of genetic algorithms for cryptanalysis is effective and what we have described an algorithm for finding the secret key is much faster than the Shamir's algorithm.

IndexTerms: genetic algorithms, cryptosystem, the system Merkley-Hellman cryptanalysi.

I. INTRODUCTION

Merkley - Hellman Cryptosystem is a public-key cipher based on the well-knownproblem knapsack [1]. Spilman first used a genetic algorithm to crack this code [2]. Severalother researchers [3,4,5] have expanded the work in this direction concentrating mainly on theinitial parameters of genetic algorithms. At the same time, in all these papers, thisproblem is solved by finding the decryption of the ciphertext without the secret key.We put the problem of computing the secret key from the public key using the attacks based on the known plaintext using a genetic algorithm. The results of such an attack can be compared with the results of the attack algorithm A. Shamir [6] verifies whether such approach provides a real advantage.

II. DENTIFIER MERKLEY - HELMAN

A. The problem of the knapsack.

Knapsack problem can be formally stated as follows. Dan backpack volume V and a set of objects, each with a volume of $\{a_1, a_2, a_3, \dots a_n\}$.Our goal is to find a subset of the set $\{x_1, x_2, x_3, \dots x_n\}$for which the equality

$$V = \sum_{i=1}^{n} x_i a_i$$

ISSN 1512-1232

where $x_i \in \{0,1\}$, for all $i = 1,2,\cdots,n$.

In general, the presence of such a subset requires exhaustive search, whose complexity is $O(2^n)$, however, there is a special case when the problem is elementary solved.

B. The calculation of the public key from the private kay.

The algorithm Merkley - Hellman as a secret key takes $\{a_1, a_2, a_3, \ldots a_n\}$sequence, where each $a_i$condition

$$a_i > \sum_{j=1}^{j-1} a_j$$

For this sequence, there is an algorithm that solves the problem of the knapsack and complexity, which is a polynomial of (or rather linear). The public key of the privation, prepared by the following transformation. Choose a number m, which satisfies the condition

$$m > \sum_{i=1}^{n} a_i$$

and the number $t < m$, wherein $(t,m) = 1$, and elements of the public key is calculated by the formula:

$$b_i = ta_i (\mathrm{mod}\, m).$$

C. Encryption and decryption.

Message is converted to a bit sequence and divided by L blocks. Each block contains exactly n bits. Calculate the sum of. $s_1, s_2, \ldots s_L$ formula

$$s_j = \sum_{i=1}^{n} x_{ij} b_i,$$

where aij is the i- the element of the j — block of the open test . $s_1, s_2, \ldots s_L$ and there is an amount - ciphertext.

To decrypt each sum is multiplied by t^{-1}modulo m

$$s_j^{'} = s_j t^{-1} (mod m)$$

and the problem is solved by using a knapsack increasing over sequence for each $s_j^{'}$.

III SHAMIR ALGORITHM

Shamir broke Merkli-Hellman algorithm which showed that it is not necessary to search for it (t_0, m_0) couple and the exact over increasing sequence, which has been encrypted. It is enough to find a$(t.m)$ and a pair of $\{a_1, a_2, a_3, \ldots a_n\}$ over increasing sequence, which is derived from the b_n sequence, via following formula $a_i = b_i t^{-1} (\mathrm{mod}\, m)$. Shamir attack us knapsack system algorithm which consists of two parts. To a whole number, for which the condition is satisfied, the $t^{-1}/_m$

value for some b_i is located in these functions minimum interval. After accumulating a certain number of points, algorithm is looking for (t^{-1}, m)pair via Diophantine approximation method, which may open the key to to calculate the secret key .

IV. GENETIC ALGORITHMS

Genetic algorithm based on the mechanism of natural selection. It was firstly proposed by Holland in 1975. The algorithm starts with a random selection of candidates' solutions (analogues

ISSN 1512-1232

of chromosomes), represented in binary strings. By applying genetic operators of selection, crossover and mutation to each generation of candidates' solutions to get a new generation of candidates solutions. Thus, these rows are a new generation average better than the previous (depending on the choice of the target , or as often referred to , a fitness function) .The most important advantage is the ability of genetic algorithms parallel to the realizations of the process of finding a solution that significantly reduces the attack.

IV. ALGORITHM

Our method of attack is quite different from the methods used in the above mentioned works. Besides, we created a genetic algorithm quite different from other genetic algorithms (different in the selection criterion and crossover process).

Our genetic algorithm is described in file "genetic2.h"created by us. In "genetic" class of the file four functions are described: the fitness function (*bool fitness(vector<populatcia>&v)*), the crossover function (*void crossover (vector<populatcia>&v)*), the mutation function (*void fitness(vector<populatcia>&v*)) and the selection function (*void selektcia(vector<populatcia>&v*)).

a) The fitness function determines the extreme increase in each member (solution-candidate) of the population transmitted to it. The fitness value of the solution-candidate in the sequence is equal to the quantity of the extremely increasing members.
b) The selection function chooses the selection-candidates, which the most fulfill the fitness function, i.e. their fitness values are higher than those of others. In case the population size is Lwe choose only L/5 solution-candidates. Exactly these solution-candidates form new generations.
c) The crossover function receives the population of the solution-candidates. From this population we choose solution-candidates with t1 and t2 numbers in pairs by means of a random generator taking into account that t1 and t2 do not coincide with each other and the used pair is not repeated. Each solution-candidate is divided in two parts (at the midpoint).
d) The mutation function changes one byte of each solution-candidate. We choose the index by the random generator and change the relevant bit, i.e. if the bit value is zero it is changed in 1, and on the contrary, if it is 1, then it is changed in zero.

Our goal is, using the above described algorithm, to find such (u,m) pair, by which we will be able to find the extremely increasing sequence by the following formula:

$b_i = a_i u (\mathrm{mod}\, m)$, where$u=t^{-1}$(3)

The algorithm is realized on C++ language base. It consists of the preparation and main parts. In the preparation part the information-to-be-transmitted is ciphered by the Merkle-Hellman algorithm. We took $\{b_1, b_2, ...b_n\}$ extremely increasing sequence, m module root and selected t multiplier, by means of which we calculated open key $a_i = b_i \cdot t (\mathrm{mod}\, m)$ and ciphered the information-to-be-transmitted by the formula (3).

The working chart of the main algorithms is as follows:

1. The initial population is represented by m root, which is initialized by random generator (it is represented in binary system). The size of each member (solution-candidate) of the population is d*n, where n is equal to the length of the open key and d=2.
   The solution-candidates are transformed into binary system.
2. Like the Shamir algorithm we take the first four members of the open key and calculate the inverse of t multiplier by m root $u = p * m / a_i$, where $u = t^{-1}$, $1 <= p <= a_{i,}$ $0 <= i < 4$.
   Thus, we receive the population all probable multipliers. We set limits for selecting u multiplier. Besides (u,m)=1and u<m, u multiplier multiplied by the third member of the

ISSN 1512-1232

open key must exceed m root. By adding this limit we reduce the solution-candidates of u multiplier, i.e. make the algorithm more purposeful. We find the relevant closed key by (3) formula for all probable candidates of (u,m)pair.

3. We determine the criterion for selection by the fitness function. In this case the criterion for selection is the extreme increase in the closed key obtained as a result of the fourth phase. In case the sequence is extremely increasing, i.e. the value of the fitness function is less than n, we pass to the following phase.
4. By means of the crossover function we carry out the crossover operation for the chosen solution-candidates.
5. For the received solution-candidates the second, third and fourth phases are repeated. In case the fitness function of any solution-candidate is equal to n, it means the desired result is obtained and the program stops functioning. Otherwise, we pass to the following phase.
6. The selection function chooses the L/5 (L is the size of the initial population)number solution-candidates, the fitness functions of which are higher.
7. We have indicated that the process will repeat 10 times. If this process is repeated, 10 times and we don't get the desired result, only in this case we use a mutation, or change the function of the gene, and then repeat the 2nd, 3rd and 4th steps. When We get the desired results, we stop working. But tests showed that non f the mutations feature is not needed, and the hybridization of a maximum of 5 times using we get the desired result table shows the results of the experiments.

	Kkey length	Population size	The number of experiments carried out	Repeating the average number of genetic operators (crossover operation)	Average execution time
1.	16	20	5	3	4.52st
2.	16	30	5	4	7.41st
3.	20	50	5	5	8.71st
4.	24	60	5	5	13.78st

According to the experiment results it is obvious that by the use of genetic algorithms the Merkle-Hellman cryptosystem is cracked quite quickly. Consequently, we may conclude that use of genetic algorithms will be useful for cryptanalysis of other asymmetric cryptosystems as well.

REFERENCES

[1] S Merkle R.C., Hellman M.E. Hidding information and signatures in trapdoor Knapsak, IEEE Trans. Inform. Theory, IT-24 (1978), pp. 535-530

[2] Spillman R. Cryptanalysis of Knapsack ciphers using genetic algorithms. Cryptologria, October, 1993.

[3] Yaseen, Sahasrabuddhe. A Genetic Algorithm for cryptanalysis of Chor Rivest Knapsack Public key cryptosystem, Third international conference on computational intelligence and multimedia applications, 1999.

[4] Garg P., Shastri A. An Improved Cryptanalytic Attack on Knapsack Cipher using Genetic Algorithm. International Journal of Information Technology 3:3 2007.

ISSN 1512-1232

[5] Muthuregunathan R., Vekataraman D., Rajasekaran P. Cryptanalysis of Knapsack Ciher Using Parrallel Evolutionary Computing. International Journal of Recent Trends in Engineering, Vol. 1, No 1, May 2009.

[6] Shamir A. A Polinomial-Time Algorithm for Breaking the Basic Merkle-Hellman Cryptosystem. IEEE Transactions on Information Theory Vol., IT-30, No5, september 1984, pp. 699-704.

Article received: 2016-11-15