Heuristic cryptanalysis of classical and modern ciphers

Conference Paper · December 2005

DOI: 10.1109/ICON.2005.1635595 · Source: IEEE Xplore

CITATIONS
5

READS
201

3 authors, including:

Ab RANI Samsudin
University of Sharjah
133 PUBLICATIONS **833** CITATIONS
SEE PROFILE

Bahari Belaton
Universiti Sains Malaysia
73 PUBLICATIONS **204** CITATIONS
SEE PROFILE

Heuristic Cryptanalysis of Classical and Modern Ciphers

Ho Yean Li, Azman Samsudin, and Bahari Belaton
School of Computer Science, Universiti Sains Malaysia
Penang, Malaysia
{yeanli,azman,bahari}@cs.usm.my

***Abstract*—Block cipher algorithms are commonly used to secure confidential information in everyday user applications However, it is quite common for ignorant users to use familiar dictionary words as their personal passwords. This research will examine the effects of weakly chosen password-keys on the security of block ciphers. A new hybrid optimization heuristic cryptanalytic attack (Tabu Search and Genetic Algorithm) is used to conduct an intelligent key-search attack on classical ciphers and modern ciphers. The algorithm chosen to represent modern block ciphers is the Advanced Encryption Standard (AES) algorithm. AES is an algebraic product cipher which combines elements of substitution and transposition. Therefore, the primarily aims of this paper is to study the effects of an optimization heuristic cryptanalytic attack on block cipher.**

***Keywords*—Cryptographic Ciphers, Cryptanalysis, and Heuristic Search.**

I. INTRODUCTION

IN today's K-Economy where knowledge means power, cryptology is an integral part of the study of securing information and preventing confidential data from falling into the wrong hands. There are two main types of cryptographic algorithms: symmetric-key and asymmetric algorithms. Symmetric-key algorithms can be divided into two categories: block ciphers and stream ciphers. Figure 1 illustrates the different classifications of Cryptographic ciphers.

This study is aimed at examining the application of a hybrid optimization heuristic cryptanalytic attack on weakly chosen keys in block ciphers; namely classical ciphers and modern ciphers. A block cipher is a symmetric-key cryptographic cipher which uses the same key to encrypt as well as decrypt fixed blocks of a secret message. There are two categories of classical ciphers: substitution ciphers and transposition ciphers. Most modern ciphers are product ciphers, which are extensions of classical ciphers. A product cipher is a block cipher which is an amalgam of components made of substitution and transposition ciphers [1].

The Advanced Encryption Standard (AES) [3] is chosen to represent the group of modern ciphers because it is a relatively new product cipher. Among all the transposition ciphers, the Columnar Transposition Cipher which is most similar to the ShiftRows step of the AES algorithm will be studied as well. Since most modern ciphers combine polygraphic substitution ciphers with a transposition cipher, the Hill Cipher is chosen to represent the substitution ciphers in this study. Furthermore, the Hill Cipher is also quite similar to the SubBytes step of the AES algorithm.

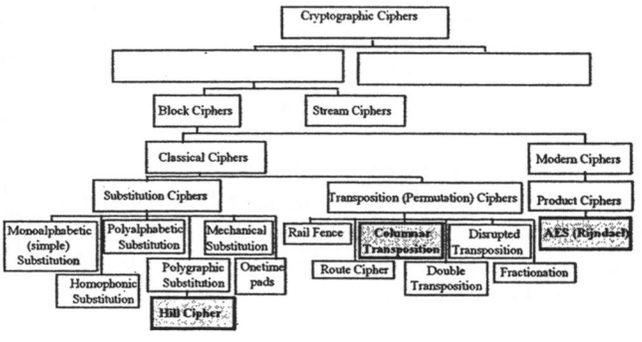

Fig.1. Schematic representation of cryptographic cipher classification (adapted from [1, 2]).

II. RELATED WORK

A. The Substitution Cipher (The Polygraphic Hill Cipher)

A substitution cipher maintains the original position of a plaintext character in the ciphertext but substitutes the value of a plaintext with another value [1]. A polygraphic Cipher substitutes blocks of characters in groups; usually pairs of characters known as bigrams. The Hill cipher is a polygraphic substitution which combines and substitutes groups of letters in a block matrix using linear algebra. According to Stallings [4], the frequency distribution of bigrams is more evenly spread in ciphers like the Hill Cipher as compared to the frequency distribution of individual letters in a monoalphabetic cipher. This makes polygraphic more difficult to break the ciphertext. It is difficult to break the Hill Cipher based on known-ciphertext only. However the linearity of the Hill Cipher makes it vulnerable to known-plaintext attacks. Hence, it is usually combined with a permutation (transposition) component as found in Modern Ciphers like Feistel Ciphers.

B. The Transposition Cipher (The Columnar Transposition Cipher)

The transposition cipher rearranges the positions of the plaintext characters in a different and complex order but "leaves the value of a character or character string unaltered

This work was supported by Universiti Sains Malaysia.

when transforming plaintext into Ciphertext" [1]. The Columnar Transposition Cipher arranges the plaintext in a square matrix from left to right and from top to bottom. It depends on the key to determine the number of columns for the letters in the square. Each character in the key becomes a column header followed by the plaintext message in successive rows beneath those headers. Spaces are ignored or replaced with a "null" value. Finally, the encrypted message is written in groups according to columns.

The transposition cipher basically rearranges the content according to a regular pattern. This could be made more complex by additional shuffling the positions of the characters.

C. The Modern Cipher (The AES Algorithm)

AES was designed to overcome the weaknesses and flaws discovered in the design of the Data Encryption Standard (DES). This improved algorithm was meant to be a replacement for DES or triple DES. In addition, the designers of AES claim that the common means of modern cipher cryptanalytic attacks are ineffective against AES due to its design structure. "However, compared to the analysis of DES, the amount of time and the number of cryptographers devoted to analyzing AES are quite limited"[4].

Although AES is an algebraic algorithm with a simple mathematical structure, it does not necessarily mean that it would be easy to break. Up till now, there are only two main directions explored in the cryptanalytic efforts on AES: the algebraic attacks on the S-box and the "Square Attack" on the key schedule.

The main trend of cryptanalytic attacks on AES is based on the Square Attack, which is considered the best known approach to attacking AES so far. The Square Attack and its subsequent variants (the Collision Attack, the Partial Sums Attack and the Related-Key Attack) are unable to break a full version of AES. The attacks only successful in reducing the complexity for about 60 – 70% of the number of rounds required for a complete AES algorithm [8]. The designers of AES have foreseen the possibility of an attack on a few rounds of AES using the Square Attack and set a very high minimum limit for the number of encryption rounds required for each key length to safeguard the security of the algorithm. The best results obtained so far are for 7 out of 10 rounds for 128-bit keys, 8 out of 12 rounds for 192-bit keys and 9 out of 14 rounds for 256-bit keys [8]. This mysterious factor (which limits the number of rounds applicable to the attacks – about 60-70%) is holding the security of AES at the moment.

So far, all the cryptanalytic attacks surveyed are impractical and insufficient to reduce the complexity of an attack on a full version of AES. Most of the attacks focus on the key or the key schedule (with the exception of the XL and XSL attacks which focus on the S-box). Although the designers of AES have made sure that an exhaustive key search on AES would be impractical, the results from the variants of Square attacks show that the complexity is significantly decreased (244 as compared to 272 for a 6-round attack on all key lengths) if combined with an intelligent key search attack.

D. Optimization Heuristic Attacks

Some intelligent cryptanalytic brute-force attacks have been conducted on cipher keys using artificial intelligent methods like simulated annealing, Genetic Algorithm and Tabu Search [1, 11-16]. So far, these search algorithms have only been attempted on classical ciphers like substitution ciphers [1, 13, 15] and transposition ciphers [1, 16] as separate entities. Nevertheless, there has yet to be an attempt to apply these algorithms to a product cipher. A modern cipher is a product cipher which is a combination of both the substitution and transposition cipher. Therefore, the results would be different because of the combination of dispersion and confusion factors involved. However, based on the statistics observed [1, 11-16], there is a good chance that there would be a general improvement in terms of search complexity as compared to an exhaustive-key search if these algorithms were applied as intelligent key-search.

According to [6], Genetic Algorithm and Tabu Search out-performed simulated annealing with positive results. The results presented in [16] also show that Genetic Algorithm and Tabu Search perform better against transposition ciphers (although the authors claim that simulated annealing is more powerful). Hence, this experiment will be conducted using a new optimization heuristic approach. The Genetic Algorithm introduces diversity into the solution pool whereas the Tabu Search prevents the same solution from being re-evaluated too soon.

Genetic Algorithms were first introduced by Holland [17] to solve problems based on the evolutionary process of gene reproduction. Figure 2 shows a general overview of the algorithm which is adapted from its biological counterpart. The Genetic Algorithm begins with a pool of pre-computed solutions (gene pool). Two solutions (parent chromosomes) with the best fitness are selected from the pool to go through the reproduction process, where specific alleles in both parents are swapped randomly to produce two new children with a combination of genes from both parents. Each child is then evaluated to determine its fitness value. The fittest child is selected for the next phase known as mutation. During the mutation phase, specific locations (loci) in the chosen individual are replaced with randomly chosen values to produce an individual with a better fitness value. The new individual is returned to the solution pool and the cycle repeats itself for the successive generations.

The Tabu Search [18] algorithm maintains several Tabu Lists to represents taboo moves in short-term and long-term memory. This algorithm is usually problem-specific. An initial solution is generated and updated with a better solution at each consecutive iteration. Long-term Tabu Lists store frequency values while short-term Tabu Lists store regency values. Aspiration criteria allow taboo moves to be executed if the overall solution is an improvement.

The structure of an English language word consists of unigrams, bigrams and trigrams. Studies have been done to determine the probability of occurrence for characters in the English language [19]. The study reveals the general order of frequency for the occurrence of each character and common bigrams and trigrams. These frequency statistics can be used to determine the probability of occurrence for

each unigram, bigram and trigram in a potential password key. Generally, vowels are the most frequently used character in the English language. A heuristic function could be created to piece together some of these elements to form a word, which could ultimately be the correct password used to form the possible key solution.

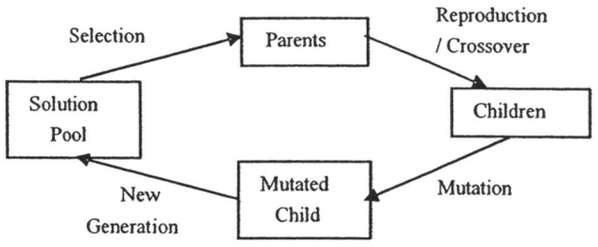

Fig. 2. The Evolutionary Process.

III. Proposed Design Methodology and Framework

A. Overall Framework of the Proposed Solution

A known plaintext will be encrypted by the chosen cipher using a randomly chosen key of reduced length. The possible key-solution generated by the heuristic function will be used to decrypt the known-ciphertext. The resulting plaintext is compared to the original. The fitness value for the solution is obtained by decrypting the known-ciphertext and calculating the percentage of character-location matches in the original plaintext and the decrypted ciphertext. The intelligent search for the correct key combination will continue until a solution match has been found or the closest match has been found within the constraints of the test environment.

For uniformity, a general structure of the proposed methodology was applied on the Hill Cipher, the Columnar Transposition Cipher and the AES. Following sections briefly illustrate a general outline of the proposed methodology for the Tabu Search Algorithm and the Genetic Algorithm. Each series of tests will consist of three trial runs of the full test cycle (one full round) to obtain the average search results of that particular test series.

In order to observe the unique properties and to allow unbiased comparisons between the three different types of cipher algorithms, a uniform structure and environment was used to conduct the tests. The following criteria of the cipher algorithms were adjusted to prepare a suitable uniform environment for testing in the limited time frame given:

1) A uniform intelligent known plaintext-known ciphertext key-search attack using Tabu Search and Genetic Algorithm was conducted on all three types of cipher algorithms.
2) The continuous tests were conducted on Pentium IV 1.50 GHz Computer with 256MB RAM running on a Linux C platform.
3) Only character-location matches will be considered. Upper hex matches and lower hex matches will not be considered for uniformity among cipher algorithms.
4) The plaintext message used for testing is limited to a standard of 16 bytes (128 bit).
5) The encryption and decryption key will be limited to a fixed maximum 8-byte English dictionary word.
6) The symmetric key will only contain English syllables and common dictionary words.
7) Only ASCII characters will be considered. This will reduce the complexity of the attack to a maximum of 56^{16} encryptions for an exhaustive key search. (This would take a maximum of approximately 2.97×10^{8} years of brute-force attack provided 1 million encryptions are done every microsecond).
8) The Tabu Search Algorithm will search randomly for possible key solutions from a pool of known words in the English language. The length of these keywords can be from 1 character to a maximum of 8 characters.
9) For the Tabu Search test run, an assumption is made that the plaintext message is encrypted with a commonly known weak password included in the pool of passwords.
10) For the purpose of comparison, the Tabu Search test will be conducted on two separate pools on different occasions. The first pool contains 2,275 common passwords (8 characters or less) in upper case, lower case and title case. The second pool contains 72,504 common dictionary words (8 characters or less) in lower case.
11) The Genetic Algorithm will be constrained to search randomly for possible 8-character key solutions from a pool of known syllables in the English language. This pool consists of 27 unigrams, 30 bigrams and 12 trigrams.
12) For the purpose of comparison and uniformity with the Tabu Search test, the Genetic Algorithm test will be conducted on two separate pools on different occasions even though the pool size has no bearing on the ultimate results. The first pool contains 320 common passwords (exactly 8 characters) in upper case, lower case and title case. The second pool contains 27,020 common dictionary words (exactly 8 characters) in lower case.

B. Proposed Tabu Search Algorithm Framework

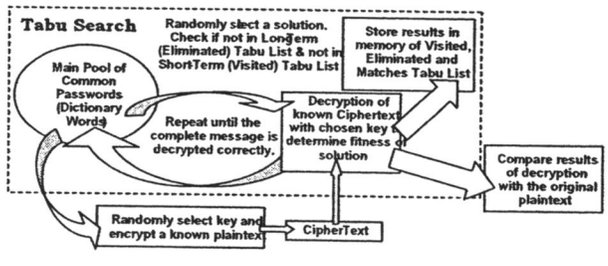

Fig. 3. One Full Test Round of *Tabu Search Algorithm*

For the purpose of uniformity, consider the encryption and decryption process as a black box. A description of a series of test rounds is as follows (summarized by Figure 3):

1) Run steps 2-5 for the Hill Cipher, the Columnar Transposition Cipher and the AES Cipher.
2) Initialize two Long-term memory Tabu Lists: "Eliminated" and "Matches". Initialize one short-term memory Tabu List "Visited" of length n/2, where n = number of solutions attempted from the solution space.

Randomly select an encryption key from the main solution pool and encrypt the known plaintext message.

3) Randomly select a solution (keyword) from the main solution pool and evaluate the fitness of the solution. Calculate the fitness value for the solution by decrypting the known-ciphertext and calculating the percentage of character-location matches in the original plaintext with the decrypted ciphertext. Store the fitness value of the current solution. If the fitness value is zero, store the solution in the "Eliminated" Long-term memory Tabu List. If the fitness value is > 0, store the matched character-location value in the "Matches" List and store the solution in the "Visited" Short-term Tabu List.
4) Repeat step 3 and compare the fitness value of the new solution with the old solution. Repeat steps 3-4 until an exact match has been found. Identify the total number of decryptions required to decrypt the full message correctly.
5) Repeat steps 2-4 twice to produce a test series of 3 test rounds. Obtain the average number of search keys required to decrypt the full message correctly.

C. Proposed Genetic Algorithm Framework

For the purpose of uniformity, consider the encryption and decryption process as a black box. A description of a series of test rounds is as follows (summarized by Figure 4):

1) Run steps 2-11 for the Hill Cipher, the Columnar Transposition Cipher and the AES Cipher.
2) Randomly select an encryption key from the main solution pool and encrypt the known plaintext message.
3) Create a new solution pool from the pool of common syllables and calculate the fitness value for all the solutions in the new solution pool by decrypting the known-ciphertext with each solution key by calculating the percentage of character-location matches in the plaintext and the decrypted ciphertext.

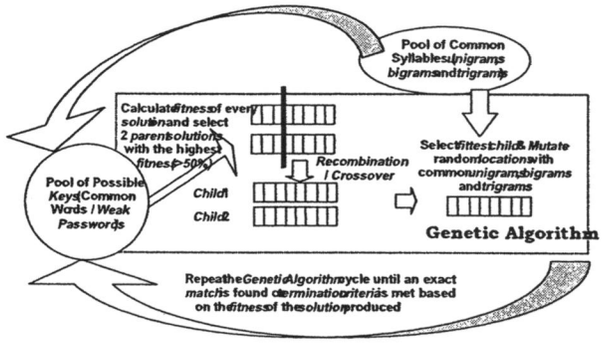

Fig. 4. One Full Test Round of Genetic Algorithm

4) Choose two solutions with the best fitness value. Each solution should minimally be able to recover at least 50% of the original plaintext message.
5) Randomly select a "crossover" point and swap the contents between the two solution key arrays.
6) Evaluate the fitness for each new child (solution key) by decrypting the known-ciphertext with each "child" key and calculating the percentage of character-location matches in the plaintext and the decrypted ciphertext.
7) Choose the "child" solution with the highest fitness value.
8) Randomly select locations and mutate the selected locations with arbitrarily chosen unigrams, bigrams and trigrams from the pool of common syllables.
9) Evaluate the fitness of the solution. If the fitness value is better than the current fitness value, update the current fitness value and the best solution variables.
10) Repeat steps 8-9 until the fitness value is 100% or there is no change in best fitness for a predetermined number of iterations.
11) Repeat steps 2-10 twice to produce a test series of 3 test rounds. Obtain the average number of search keys required to decrypt the full message correctly.

IV. Implementation and Results

A. Implementation Problems

The total amount of time needed to get the results for intelligent key-search attack depends on three major factors: the probability of random selection, the weakness of the keyword chosen and the strength of the cipher structure against a heuristic attack. The Genetic Algorithm proved to be most efficient on transposition cipher (the Columnar Transposition Cipher). However, it was also observed that the Genetic Algorithm produced weak results for the substitution cipher (the Hill Cipher) and the modern cipher (the AES product cipher).

After many trial runs, it was discovered that the processing power of the test environment was insufficient to completely recover the full plaintext message from these two ciphers (the Hill Cipher and the AES product cipher). Nevertheless, the attacks were successfully conducted on the full-cycle versions of all the cipher algorithms to produce measurable results.

B. General Findings: Results of Implementation of *Proposed* Tabu Search *Algorithm Framework*

Figure 5 and Figure 6 summarize the results obtained from 21 test runs (seven series) of the Tabu Search algorithm on the three types of ciphers (AES, Hill and Columnar Transposition). Result from Figure 5 was based on a pool of 2,275 possible keywords and result for Figure 6 was based on a pool of 72,504 possible keywords. Figure 7 to Figure 9 shows the effectiveness of Tabu Search on the ciphers based on the pool size comparison.

C. General Findings: Results of Implementation of *Proposed* Genetic Algorithm *Framework*

For uniformity, the Genetic Algorithm is tested using an encryption key from two pools. However, the pools of encryption keys only contain keywords which are exactly 8 characters long. The sizes of the two pools are 320 keys and 27,020 keys respectively. Overall, the Genetic Algorithm produced results from the Columnar Transposition Cipher fast and efficiently. In fact, the performance against this cipher was better than the Tabu Search. However, the Genetic Algorithm generally did not perform well on the other two ciphers, namely the Hill Cipher and AES. In most of the cases, the Genetic Algorithm could not produce any significant positive result from these two ciphers at all. After one month of continuous test runs, it was discovered that these two ciphers have a consistent pattern: One test run

cycle can last up till 8-12 hours before the computer fails and crashes in the midst of building the initial solution pool. Consequently, an important point to note is that when attempts were made to use the Genetic Algorithm on the Hill Cipher or the AES Cipher, the process almost never goes beyond the first step of initializing the solution pool and obtaining two parent key solutions with a minimum fitness of 50% or more.

Figure 10 summarizes the average results of conducting 10 series of test runs (total of 30 test runs) of the Genetic Algorithm on the Columnar Transposition Cipher.

Figure 11 illustrates a pattern, showing the relationship between the total numbers of generations of key solutions required to be tested before an optimal solution is found vs. the initial pool size required to obtain two parents with a minimum fitness of 50%.

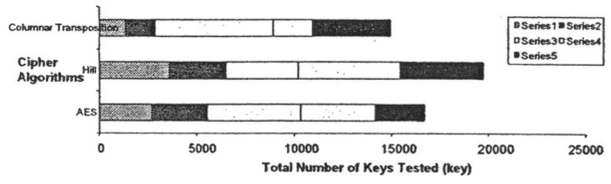

Fig. 5. Comparison of Effectiveness of Tabu Search on Cipher Algorithms: 15 rounds of Tabu Search (Tabulation Based on Total Search Keys Required in Pool Size of 2,275 words)

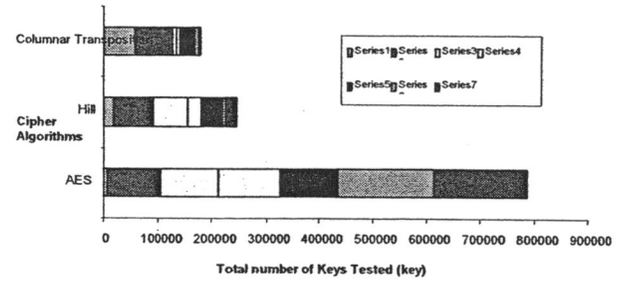

Fig. 6. Comparison of Effectiveness of Tabu Search on Cipher Algorithms: 21 Test Rounds of Tabu Search (Tabulation Based on Total Search Keys Required in Pool Size of 72,504 words)

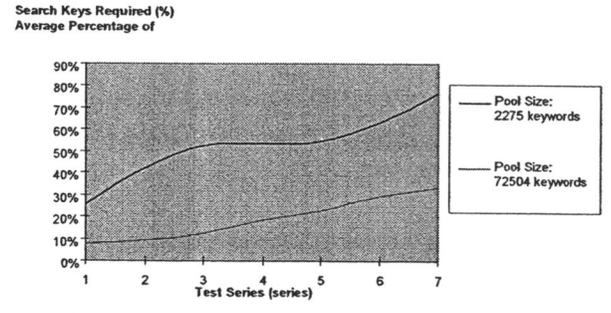

Fig. 7. Effectiveness of Tabu Search on Hill Cipher: A Comparison Based on the Keyword Pool Size

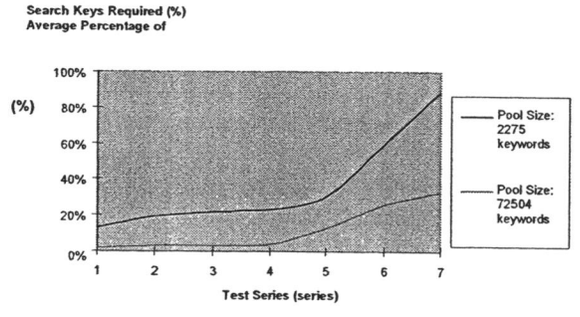

Fig. 8. Effectiveness of Tabu Search on Columnar Transposition Cipher: A Comparison Based on the Keyword Pool Size

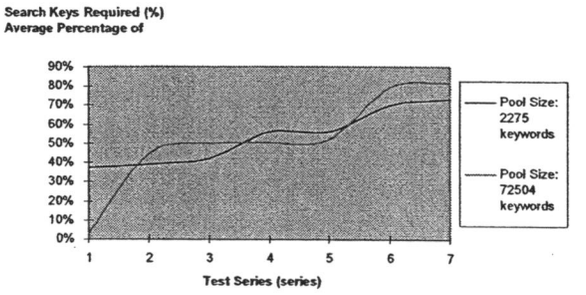

Fig. 9. Effectiveness of Tabu Search on AES Cipher : A Comparison Based on the Keyword Pool Size

D. Discussion of Results

Proposed Tabu Search Algorithm Framework

A trend was observed from the results of this research that regardless of the strength of the cipher algorithm, the performance of the Tabu Search attack is generally improved if the attacker uses a larger pool of known potential weak password. However, contrary to the characteristics of the two classical ciphers the security of the AES cipher proved to be relatively stable and did not vary too much with the change of Tabu Search keyword pool size. Although there is a very slight improvement in the performance of the Tabu Search attack on the AES Cipher with the increase of the potential keyword solution pool size, the changes are very minor and almost negligible.

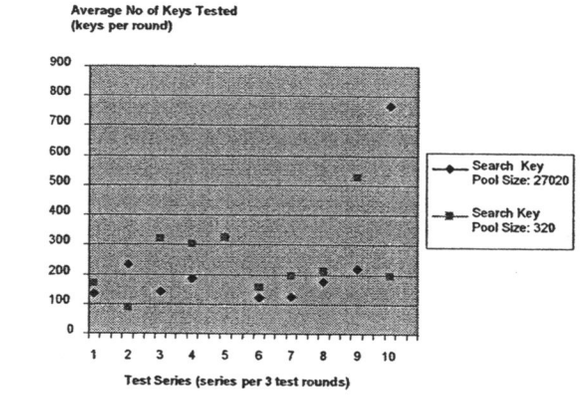

Fig. 10. Effectiveness of Key Search Using Genetic Algorithm on Columnar Transposition Cipher (10 series – 30 test rounds)

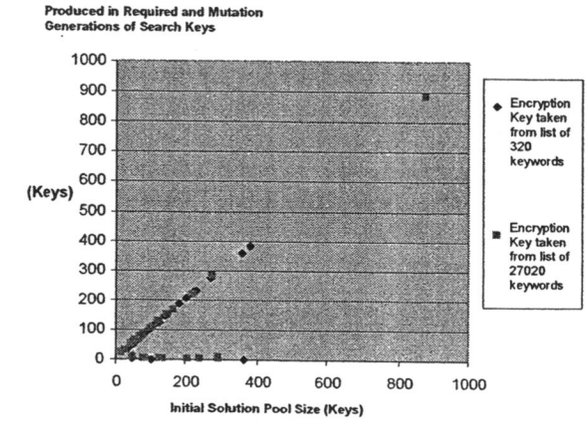

Fig. 11. Genetic Algorithm on Columnar Transposition Cipher: Relationship between the Total Generations of Solutions Required and the Size of the Initial Solution Pool Generated

TABLE 2
SUMMARIZED AVERAGE RESULTS FOR THE APPLICATION OF *GENETIC ALGORITHM* ON THE *AES CIPHER*

	Fitness (Percentage of Plaintext Recovered)	Initial Pool Size (keys)	Generations (keys)	Estimated Average Time Span to Obtain Best Parent Solution (Hours)
Best Case	25.0%	18580088	0	4.5
Average	12.5%	385592	0	2.5

This is surprising considering the AES cipher is a product cipher which should contain the properties of both the substitution and the transposition ciphers. It appears as if the product cipher has inherited more of the strengths of both types of classical ciphers but very little of the weaknesses. However, this proves that although the strength of the AES product cipher is affected by the strength of the key to a certain degree, the cipher's security is relatively stable because it does not fully depend on the security of the key alone.

Proposed Genetic Algorithm Framework

Generally, the Genetic Algorithm attack proved to be most efficient against the transposition cipher. The attack succeeded in recovering the original plaintext message in less than an hour for each trial run. Nonetheless, it was also observed that the Genetic Algorithm attack produced weak results for the substitution cipher (the Hill Cipher) and the modern cipher (the AES product cipher).

This is due to the fact that the original plaintext message may be recovered by using an alternative key with similar properties as the original encryption key on the transposition cipher, but never on the substitution cipher or on the product cipher. This is because of the confusion property inherent in both the substitution cipher and the product cipher. Generally, the results suggest that a parallel implementation of the Genetic Algorithm would produce better results than the serial implementation done here.

V. CONCLUSION

The results have shown that the transposition cipher (Columnar Transposition Cipher) is most susceptible to the Tabu Search and Genetic Algorithm attacks on weak passwords. This is followed by the Polygraphic Substitution Cipher (Hill Cipher), which is also vulnerable to the Tabu Search attack as well as the Genetic Algorithm attack, but at a greater time cost (provided the encryption key is a weakly chosen password). The product cipher (the AES cipher) is the most secure among the three. Unlike the other two, the product cipher is rather stable in terms of its vulnerability towards the optimization heuristic attacks. Nevertheless, the product cipher is still susceptible to weak password attacks by the average hacker or script kiddie using a basic personal computer system. This is especially obvious from the average 52% - 53% key search efficiency using the Tabu Search algorithm. In short, regardless of the strength and security of a cryptographic cipher, all categories of cipher algorithms are vulnerable to optimization heuristic attacks by a basic personal computer if the encryption key is a weakly chosen password.

REFERENCES

[1] Gründlingh, Werner R. and Van Vuuren, Jan H. *Using genetic Algorithms to Break a Simple Cryptographic Cipher.* http://www.apprendre-en-ligne.net/bibliotheque/genetic.ps

[2] Schneier, Bruce. 1996. *Applied Cryptography - Protocols, Algorithms, and Source Code in C.* Second Edition. Canada: John Wiley & Sons.

[3] National Institute of Standards and Technology, U.S. department of Commerce. November 26, 2001. *Advanced Encryption.*

[4] Stallings, William. 2003. *Cryptography and Network Security Principles and Practices.* Third Edition. New Jersey: Prentice Hall. pp.29,37-40,653.

[5] *Standard(AES).* FIPS PUB 197. http://csrc.nist.gov/publications

[6] Courtois, N.T. and Pierprzy, J. Dec 2002. *Cryptanalysis of Block Ciphers with Overdefined Systems of Equations.* Asiacrypt 2002.

[7] Moh, T. September 18, 2002. *On the Courtois-Pieprzk's Attack on Rijndael.* http://www.usdsi.com/aes.html

[8] Murphy, S. and Robshaw, M.J.B. 2002. *Essential Algebraic Structure within the AES.* Crypto 2002. http://www.isg.rhul.ac.uk/~mrobshaw/rijndael/aes-crypto.pdf

[9] Ferguson, Niels; Kelsey, John; Lucks, Stefan; Schneier, Bruce; Stay, Mike; Wagner, David and Whiting, Doug. 2000. *Improved Cryptanalysis of Rijndael.* Springer-Verlag. http://www.macfergus.com/pub/ icrijndael.pdf

[10] Babbage, Steve. November 11, 2002. *Rijndael and other block ciphers.* NESSIE Discussion Forum. http://www.cosic.esat.kuleuven.ac.be/nessie/forum/read.php?f=1&i=82&t=82

[11] Dlanielyan, Edgar. February 2001.*AES:Advanced Encryption Standard is Coming.* ; login:, the magazine of USENIX and SAGE 26(1): 62.

[12] Kolodziejczyk, Joanna. 1997. *The Application of Genetic Algorithm in Cryptanalysis of Knapsack Cipher.* Proceeding of European School on Genetic Algorithms. Eurogen '97. http://ingenet.ulpgc.es/functional/eurogenxx/eurogen97/contributed/kolodziejczyk/ht/kolodziejczyk.htm

[13] Spillman, Richard. October 1993. *Cryptanalysis of Knapsack Ciphers using Genetic Algorithms.* Cryptologia XVII (4). pp. 367-377. http://www.plu.edu/~janssema/ga_solve.zip

[14] Clark, Andrew and Dawson, Ed. 1998. *Optimisation Heuristics for the Automated Cryptanalysis of Classical Ciphers.* Journal of Combinatorial Mathematics & Combinational Computing. Vol 28. pp.63-86. http://sky.fit.qut.edu.au/~clarka/papers/jcmcc1998.pdf

[15] Lebedko, O. and Topehy, A. 1998. *On Efficiency of Cryptanalysis for Knapsack Ciphers.* Poster Preoceddings of ACDM'98 PEDC. http://www.msu.edu/~topchyal/acdm98.ps

[16] Dimovski, A. and Gligoroski, D. October 2003. *Attacks on the Transposition Ciphers UsingOptimization Heuristics.* Proceedings of ICEST 2003. http://www.pmf.ukim.edu.mk/~danilo/ResearchPaper/Crypto/AttackTranspositionICEST2003.pdf

[17] Dimovski, A. and Gligoroski, D. March 2003. *Attack On the Polyalphabetic Substitution Cipher Using Genetic Algorithm.* Technical Report, Swiss-Macedonian scientific cooperation trought SCOPES project. http://www.pmf.ukim.edu.mk/~danilo/ResearchPapers/Crypto/AttackPolyalphabeticSCOPES2003.pdf

[18] Holland, J. 1975. *Adaptation in Natural and Artificial Systems.* Ann Arbor, Michigan: University of Michigan Press.

[19] Glover, Fred; Taillard, Eric; and de Werra, Dominique. 1993. *A user's guide to tabu search.* Annals of Operations Research, 41. pp. 3-28.

[20] Crypto'04. 2004. *Most Common Letters, Digrams, and Trigrams in the English Language.* http://academic.regis.edu/jseibert/Crypto04/Frequency.pdf