Pertanika J. Sci. & Technol. 26 (3): 1095 - 1110 (2018)

PERTANIKA JOURNALS

SCIENCE & TECHNOLOGY

Journal homepage: http://www.pertanika.upm.edu.my/

Distance Correlation between Plaintext and Hash Data by Genetic Algorithm

Farjami, Y.*, Rahbari, D. and Hosseini, E.

Department of Computer Engineering and Information Technology, University of Qom, Iran

ABSTRACT

The hash function is used as a one-way cryptography method for digital signature and message authentication. Hash values are provided using a mathematical and logical process, so they are different from the generators of random numbers. The position analysis of bits in plaintext and its hash is very suitable to show their relationship. The focus of this paper is to point to the best relations between the plaintext and hash bits, in which the difference between hash methods will be proven. In this work, we use distance correlation (dCorr) as a measurement function of precision statistical dependency between two vectors. The genetic algorithm (GA) is used to find a set of optimal positions between plaintext and its hash data with maximum dCorr. The results of the experiment regarding dCorr indicate that MD5 as a message digest method is different from the random function. Also, the proposed method compared with Tabu search (TS) and Simulated Annealing (SA) algorithms has a lower average execution time for 1000 pairs of plaintext and hash data.

Keywords: Distance correlation, genetic algorithm, hash, message digest

INTRODUCTION

The advancement of science and technology has put security and privacy issues in network communications in the spotlight (Zhang et al., 2014; Zhang & Xiao, 2013). Messages are always subject to attack and access when they are transmitted on the network. Message authentication can be used to resist the attacks (Reshef et al., 2011). Hash is a one-way method for producing fixed-length values (Schneier, 2004) that is used for security and data availability in communication systems. Strong hash functions do not produce the same output values for different inputs, but some cases point to the strength algorithms that challenge the hash functions (Kwok, 2005). Other applications of hashing are approximate matching in forensics system (Zhai, 2017;

ARTICLE INFO

Article history:
Received: 20 September 2017
Accepted: 22 March 2018

E-mail addresses:
farjami@qom.ac.ir (Farjami, Y.)
d.rahbari@stu.qom.ac.ir (Rahbari, D.)
e.Hosseini@stu.qom.ac.ir (Hosseini, E.)
*Corresponding Author

ISSN: 0128-7680 © 2018 Universiti Putra Malaysia Press.

Choi et al, 2017; Choi et al 2016). In these systems, user assertion is recognised by comparing two hash files. The file format can be text, audio, and video. If an attacker can find two different files with the same hash, then the security of hashing method is compromised.

The dCorr is a recent method for finding the correlation between data with any size of dimension (Kwok, 2005). The Pearson correlation coefficient (PCC) generally cannot detect nonlinear relations and its value often is zero for dependent variables, thus a new measure is introduced called dCorr coefficient to address the shortcomings of the Pearson coefficient (Kwok, 2005). Moreover, unlike the Pearson coefficient, dCorr coefficient value is zero if and only if the variables are independent (Székely & Rizzo, 2009).

In this paper, we show best relationships between plaintext and its hash data with maximum dCorr by GA; the latter is a metaheuristic method with the natural selection process that is used to find optimal solutions (Mitchell, 1998). Based on dCorr, the correlation between plaintext and its hash data is calculated in certain positions, also GA finds best positions with maximum dCorr. The key contributions of this paper are as follows: 1) Prove the existence of a logical process for calculating MD5-based hash unlike the random function; 2) Find the maximum dCorr value and best relationship positions between plaintext and its hash data; and 3) Comparison of our proposed algorithm with TS and SA algorithms is based on average execution time.

The rest of the paper is organised as follows. The dCorr function calculates the relationship between plaintext and its hash by MD5 and random methods. The details of the GA are presented for a quick and optimal calculation of dCorr. Findings are discussed in the results and discussion section especially comparing the proposed algorithm with TS and SA. The final section concludes the paper.

LITERATURE REVIEW OF RELATED WORK

In Lipowski and Lipowska (2012), the authors proposed a cyber black box system for analysis of MD5 (Rivest, 1992) and SHA256 hashing methods on the network packets. They recognised transmission packets in real time and optimised their method by logging and storing of network traffic and related information generation. Traffic analysis in mobile networks as an excellent source of knowledge is a popular research field (Naboulsi et al., 2016; Martínez-Gómez et al., 2014). Detecting the relationship between data and breaking the hash is one of the important steps to find encrypted information. The related works are categorised as Maximal Information Coefficient (MIC), Distance correlation (dCorr), and other correlation measures.

Maximal Information Coefficient (MIC)

Reshef et al. (2011) used MIC to compare two random variables which is based on the mutual information with an easy implementation and strong against noise (Székely & Rizzo, 2013). Linge et al. (2014) showed how MIC applies in the particular case of the side channel attacks. They offered ways to compute the MIC with low complexity. The results show high performance of this approach when the leakage is noisy and poorly modelled (Linge, 2014).

In Gorgens et al. (2017), the airborne laser scanning was introduced for computation of height and density metrics. The MIC will be used in this study to find a relationship between forests attributes. In Wang et al. (2018), an intelligent MIC (iMIC) is offered for quadratic optimisation to create a finer normal matrix. The authors used 169 data set indices from 202 member countries of World Health Organization (WHO) to discover the local optimal value by iMIC with lower iterations and fine performance. Since the material on the y-axis is a branch of quadratic optimisation with the same partition on the x-axis, iMIC method is a better alternative than the proposed searching method by Reshef et al. (2011) In (Shao & Li., 2016), authors created the graph model with railway nodes that shows influencing factors of accidents by edges between each pairs of nodes. They proposed MIC method to discover the relationship among railway nodes to prevent accidents.

An MIC-based approach was proposed in Sun et al. (2017) where entropy epistasis was used to measure the level of factor interactions in a constant optimisation. Shuliang et al. offered an improved estimate algorithm for MIC, named the IAMIC. Their method has the ability to search for extreme values using few iterations. Reshef et al. (2014) provided a comparison between the improved algorithm and the original MIC algorithm. The MIC has lower power than dCorr-based methods (Wang et al., 2017; Székely & Rizzo, 2009) in many relationships, and it is claimed that this fault could cause MIC to lose its advantage for general use.

Distance Correlation (dCorr)

In Wang and Zhonghua (2017), dCorr is used for outlier recognition in high-dimensional regression problems and leave-one-out method. The researchers used the bootstrap method and a threshold rule to determine whether an individual is an outlier. The Monte Carlo simulations with a real data show the offered outlier recognition method can detect outlier survey and enhance the filtering accuracy in attribute hide problem.

In Górecki et al. (2017), the dCorr and canonical correlation coefficient with Fourier series are used to determine the relationship between daily temperatures for each day of the year and average daily rainfall data in Canada. The results show the values of dCorr coefficients are independent of size but the value of canonical correlation coefficient increases a little with the size of the regularisation. As a result, these methods are not suitable for univariate data. As a second experiment, they used multivariate data obtained from Professor R. Nadulski of Department of Food Machinery and Engineering. They used permutation tests that included p-values equal to zero for both methods of infrigidation (the process of cooling or freezing for preservative purposes) and all basis sizes. The authors suggested dCorr to find correlation relations.

Wang et al. (2016) studied the intrusion detection and variable screening in large-scale regression. The dCorr was used to discover and analysis correlation of actual data with high dimensional. They found the correlation between responses and predicted data with dCorr and pointed that the proposed method can reduce unfavourable effects after removing powerful data tips. The filtering accuracy can be refined in variable screening problem. In (Pan et al., 2017), the authors established conditional local dCorr coefficient to determine a nonlinear relationship between multivariate data. In Wang et al. (2017), the dCorr method and G2 statistic are used for Bayesian modelling approach, so the authors analysed and compared the dependence and independence of nonparametric data. In Heller et al. (2016), the data space is divided into 2 * 2 tables to compare the dependencies and independence of the parts. Their strategy is similar to Fisher’s exact scanning (FES) method based on the dCorr approach for 2 * 2 spaces.

The dCorr coefficient which has more information compared with Pearson coefficient (Martínez-Gómez, 2014) is used to show nonlinear associations by variables with any dimension, and it’s possible to determine mini sets of variables that give same astrophysical information. For experiments, all values are given for Pearson’s correlation coefficient. As a result, dCorr method is stronger than the MIC (De Siqueira Santos et al., 2013) to detect astrophysical information into most concentrated horseshoe or V-shaped, and also to increase the power of classification and pattern recognition (Linge, 2014).

Other Correlation Measures

In Budhathoki and Vreeken (2017), the Minimum Description Length (MDL) method programmed in the Python environment is used to discover related variables in data analysis. In this work, the Kolmogorov complexity is applied to compact and find a correlation. The MDL model shows data correlation with high significant power, especially for binary data but this method can’t easily be extended to other data types. In another work, Exceptional Model Mining (EMM) was used to find a linear relation between subgroups of the dataset whose attributes have an unusual relationship. In Downar and Duivesteijn (2017), experiments were conducted on several datasets as Windsor Housing, the South African Heart Disease Study, the Ozone, Iris, and the Higgs Boson ML challenge. The results of correlation quality methods show that the Filler–Spearman and the Filler–Kendall rank overlap with Pearson’s correlation. The authors proposed the dCorr and MIC approach to find a strong relationship between data and subgroups of data.

PROPOSED METHOD

Distance Correlation

The dCorr takes values in [0, 1] and is equals to zero if and only if independence holds. It is more general than the classical Pearson product moment correlation. It provides a scalar measure of multivariate independence that typifies independence of two random vectors. It is well defined in both the population and in the sample and it is easy to compute (Geerligs & Henson, 2016; Székely & Rizzo, 2009). Given random variables X, Y € R let X’Ý and X’’, Y’’ be independent pairs of random variables taken from the same joint distribution as that of X and Y. The distance covariance of X and Y is defined as the square root of Eq. (1).

$$dCov^2(X,Y) = E[|X - x'||Y - Y'|] + E[|X - X'|]E[|Y - Y'|] - E[|X - X'||Y - Y''|] - E[|X - X''||Y - Y'|] = E[|X - X'||Y - Y'|] + E[|X - X'|]E[|Y - Y'|] - 2E[|X - X'||Y - Y''|] \quad (1)$$

The distance variance and dCorr are defined as Eqs. (2) and (3).

$$dVar^2(X) = dCov^2(X,X) \tag{2}$$

$$dCor^2(X,Y) = \frac{dCov^2(X,Y)}{\sqrt{dVar^2(X)}\sqrt{dVar^2(Y)}} \tag{3}$$

Algorithm Structure

The calculation of dCorr between two vectors is simple when their positions are given. The purpose of this research is to find the best positions of the plaintext and its hash data with maximum dCorr value, which requires a lot of computing.

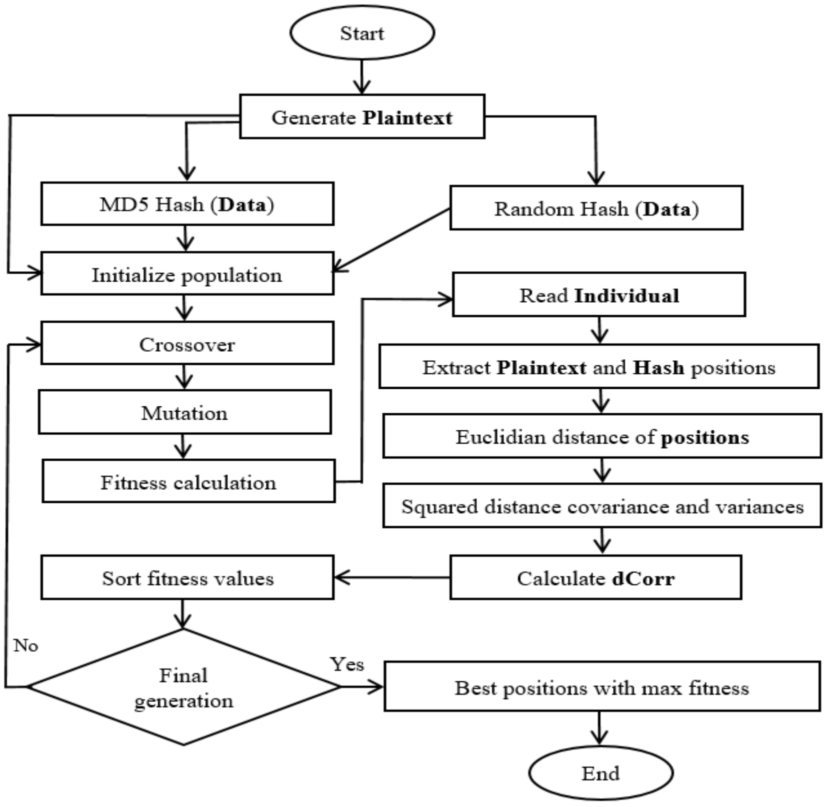

Figure 1. Flowchart of proposed algorithm

The GA is based on the idea of evolution and it can find best solutions after running several generations. GA searches the answer space with a directed random method. Flowchart of our proposed algorithm is shown in Figure 1. We generate a plaintext (P) then calculate its hash value (H) by MD5 and random methods. The pair of (P, H) sends to GA and maximum dCorr value and best bit positions of P and H are calculated. Details of the proposed method are as Algorithm 1.

Algorithm 1. **Proposed Algorithm**

```
1:   For dc = 2 to 32 do
2:       For t = 1 to 1000 do
3:           Generate plaintext as P.
4:           H1 = MD5 (P).
5:           H2 = RandomHash (P).
6:           [dCorr1, BestPositions1] = GA (P, H1, dc).
7:           [dCorr2, BestPositions2] = GA (P, H2, dc).
8:           Print best positions of P, H1, and H2 with Maximum dCorr value.
9:       End For
10:  End For
```

Our proposed method as Algorithm 1 is executed by different numbers of dCorr positions as dc in line 1. We also use this parameter for analysis of outputs. For example, to calculate the dCorr value between 2 positions of plaintext and its hash data, dc is equal to 2, also for 8 positions, dc is equal to 8. The algorithm is run by 1000 pairs of plaintext and their hash in line 2. In Line 3, a random plaintext as P with 32 characters is generated. The MD5 and random hash values of P are calculated as H1 and H2 in lines 4 and 5. The GA is called twice to find the maximum dCorr value and best positions of bits between P and H1 (MD5) in line 6, also between P and H2 (Random) in line 7. The best answers are printed in line 8. The steps of the GA as Algorithm 2 include population initialisation, crossover, mutation, selection, fitness evaluation in a certain number of iterations. These steps are explained below:

Population

The structure of a chromosome or an individual is based on a binary string, so that it has fast speed in GA operations and converts to the decimal bit positions. The string length is based on the number of dCorr positions. For example, 2 positions in plaintext and 2 positions in its hash require 32 bits in the chromosome. Because each position is a decimal value or a binary string of 8 bits. Thus, by multiplying the number 4 by 8, the string length is 32 bits, so an individual has 32 bits or genes. Additionally, for 3 positions of plaintext and its hash data, an individual has 48 bits, and so on. In the model, the first half of each individual is considered for the plaintext and the second half belongs to its hash data.

Decimal positions →	5	8	17	57
Individual →	00000101	00001000	00010001	00111001

Figure 2. One chromosome or an individual of population

We compute the main position of bits in the plaintext or hash by dCorr-based fitness function. As shown in Figure 2, by separating every 8 bits in the chromosome and converting them to a decimal value, positions 5 and 8 in the plaintext, as well as 17 and 57 in its hash data are obtained.

Crossover

In GA, the crossover operator is important for convergence towards the solutions. In Figure 3, a single-point crossover is used so that the second half of the two parent chromosomes are exchanged. The high rate of crossover operator increases the convergence speed. The parent selection stage of crossover operator is performed using roulette wheel method (Lipowski & Lipowska, 2012).

Before	Parent 1	0000010100001000 0001000100111001
	Parent 2	1001000010010001 1101010011111010
After	Child 1	0000010100001000 1101010011111010
	Child 2	1001000010010001 0001000100111001

Figure 3. The parents and childs before and after crossover

In roulette wheel, all individuals have a fitness value titled as p_i, which is divided by the sum of the fitness values of population, then a random value as r is generated for each individual. If p_i value is greater than r, then the chromosome is selected as the parent for the crossover operation. If the fitness value of two child chromosomes is better than rest individuals in the population, they will be selected to produce the next generation.

Mutation

As Figure 4, the mutation operator changes a bit of population to include plaintext and its hash positions from 0 to 1 or vice versa. The diversity rate of the population is proportional to the probability of the mutation operator rate. In fact, if we assume that the population size is 10, with a mutation rate of 30%, then the number of selected chromosomes for the mutation operation is equal to 3. After mutation operations, the plaintext or its hash positions are changed and then added to the population. If the fitness of the mutated chromosome is better than rest individuals, then it can be selected for the next generation of the GA.

Before	00000101000010000001000100111001
After	00000101000010000001001100111001

Figure 4. A chromosome before and after the mutation operator

Fitness Function

The fitness function is based on dCorr as Eq. (3). First, the plaintext and its hash data are read as an individual. Second, the plaintext and its hash positions are extracted as X and Y vectors. Third, dCorr (X, Y) is calculated. Since the purpose of the GA is to find the maximum dCorr value, thus the fitness values are in descending order. The GA is presented as Algorithm 2.

Algorithm2. ***GA***

```
Input: (P = plaintext, H = Hash, dc = dCorr positions).
Output: dCorr value and best positions.
1:  Initialise Population_size, Max_Iterations, Crossover, and Mutation probability.
2:  L = dc * 8 * 2.
3:  For t =1 to Population_size do
4:      Generate L bits for each individual.
5:  End For
6:  For i =1 to Max_Iterations do
7:      Popc = Crossover two chromosomes with roulette wheel selection.
8:      Popm = Mutation population.
9:      Replace Popc and Popm by worst individuals.
10:     Fitness calculation (dCorr):
11:         Read an individual.
12:         [Plaintext_pos, Hash_pos] = Extract positions of P and H.
13:         Calculate $X = P(Plainttex\_pos)$ and $Y = H(Hash\_pos)$.
14:         Calculate the Euclidian distance for P and H.
15:         Calculate squared distance covariance and variances.
16:         Calculate dCorr as Eq. (3).
17: Sort Fitness values.
18: End For
19: Return dCorr and best positions.
```

In Algorithm 2, the initial values of the parameters are presented in line 1 by the number of population individuals, the rate of mutation, and crossover operators. In line 2, the length of each individual is calculated as L. In the calculation of L, **dc** is the number of positions, **8** is the number of bits for a character, and **2** is for two inputs as plaintext and its hash. For example, we want to calculate the dCorr value for 4 positions of P and 4 positions of H, thus dc is equal 4. As Algorithm 1 (line2), the plaintext is 32 characters and each character is 8 bits then 32 * 8 = 256 bits or positions, thus, we need to **8** bits for addressing to 256 positions. Also, because of the two different sets of positions for P and H, we multiply (**dc*8**) to **2**. The main circle of GA is executed from line 6 to line 18. The crossover and mutation operators are presented in lines 7 and 8. After these operations, the worst individuals in the population are replaced by new child individuals in line 9. The fitness function as dCorr is calculated by lines 10 to 16. An individual include P and its H positions is read in line 11 and is extracted in line 12.

Based on the above example, we proof the decimal values of an individual are (5, 8, 12, 23, 2, 9, 16, and 28). Thus, values (5, 8, 12, and 23) are for P positions and values (2, 9, 16, and 28) are for H positions. Algorithm 2 in Line 12, gain the corresponding bit values in P and H. In line 13, two vectors are gained for dCorr calculation. The lines 14 to 16 calculate dCorr using the Euclidian distance, squared distance covariance, and variances as Eq. (3). In line 17, the fitness values of all individuals are descending. The GA is finished in line 18. In line 19, the maximum fitness value (dCorr) and best relation positions between P and H are returned to Algorithm 1.

RESULTS AND DISCUSSION

The proposed algorithm is implemented by Matlab software on a PC with CPU of Intel Core i5 2.67 GHz, 3.0 GB of RAM in Windows 10 operating system 32 bits. The selected hash functions for result evaluation are MD5 and random methods. At the start of the GA, some parameters are initialised, so that the number of genes of each chromosome is equal dc*16 as proposed algorithm description, the maximum number of iterations is equal 50, the population size is equal 10, the crossover operator probability is equal 0.8, the mutation operator probability is equal 0.3. The dc is a number of positions that is used to compute dCorr value between plaintext and its hash data.

Table 1
A sample input and output of the GA

Plaintext	Hash	P positions	H positions	dc
'Password123*()'	**MD5:**35acc07d9d81171d94afff007b2a69fa	4, 8, 12, 14	15, 23, 26, 27, 32	6
	Random:/Ñ&M_m□ Îa0Ìd)ë¡T² ìúAZáØÍí³	1, 5, 7, 8, 9	4, 7, 12, 13, 18, 27	

The number of dCorr position is equals to 4, so that the GA ultimately achieves 4 best positions. As Figure 5 (A), GA reaches to optimised answer in the fourth generation, thus the proposed method is very suitable for calculating of dCorr. Figure 5 (B) shows the values of dCorr between the plaintext and its hash in different generations of the GA, so that these values are calculated after 14 generations of the GA.

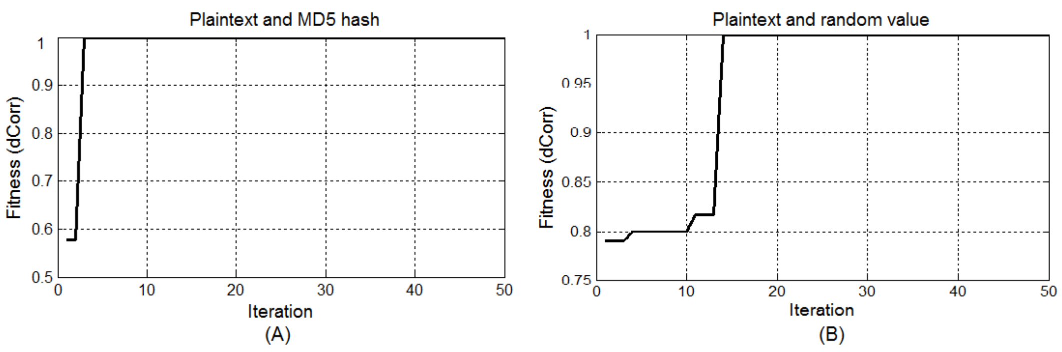

Figure 5. GA output for one plaintext with MD5 Hash in (A) and Random Hash in (B)

According to the results, the convergence rate and dCorr values of MD5 is less than the random function and the proposed algorithm finds the best MD5 dCorr values at the primitive generations. The diagrams in Figure 5 are based on the plaintext (P) and its hash (H) of Table 1. The proposed method is tested by 1000 pairs of plaintext (Each plaintext generates by 32 random characters) and their hash data by MD5 and random methods. This test is based on the different number of dCorr positions, so that their average values are given in Figure 6.

The average dCorr values for 1000 plaintexts and hash data are different in MD5 and random hash methods as seen in Figure 6. The values in horizontal axis represent dCorr value for certain position count of plaintext and its hash data. The MD5 hash function has a known process and it isn't based on random process. As per Figure 3, for positions greater than 3, dCorr values for MD5 are more than the random values. As the first result of this diagram, MD5 hash, unlike the random function, is based on a mathematical and logical process. In second, there are correlations between the plaintext and its MD5 hash data in different positions, thus, attack to MD5 hash function is feasible. If a relationship is found between the plaintext and its hash, then other relationships will be found, which will result in the hash method longer having security.

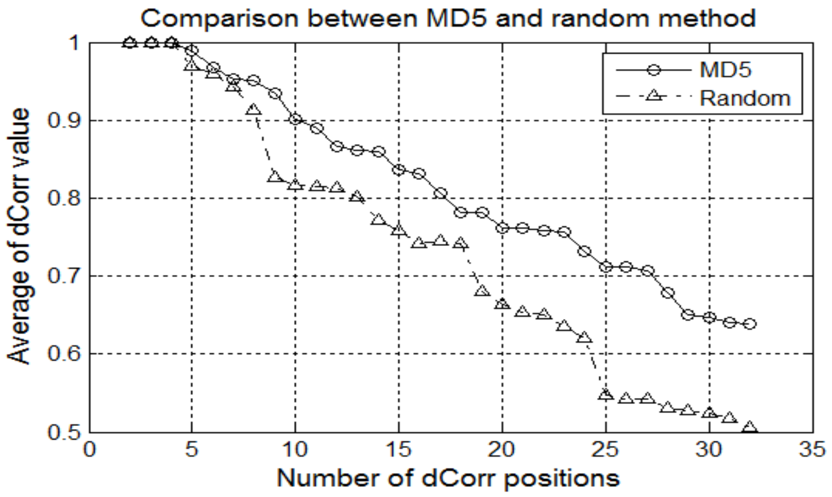

Figure 6. Comparison of dCorr for MD5 and Random Hash

The hash operation should be a one-way function but the MD5 has shown that it does not have enough security and it is not suitable for use. The main purpose of this paper is to calculate dCorr between plaintext and its hash data as well as the proof of difference with the random method. Table 2 shows different plaintext (P) and its hash (H) data as well as their best relationship positions in both MD5 and random methods, which are obtained from the GA.

Table 2
Details of GA output

Plaintext	Hash	dc	P Positions	H Positions	dCorr
‘password’	**MD5:** 5f4dcc3b5aa765d61d8327deb882cf99	2	3, 6	9, 19	1
	Random: IþRq($"dxÃ 2□§¨@8Àù½'v	7, 8	3, 30	0.99	
‘12345678’	**MD5:** 25d55ad283aa400af464c76d713c07ad	3	4, 7, 8	1, 10, 29	1
	Random: AùnÃÃ¶Î□ò¡@/«;\Õ¸Ù		2, 4, 8	7, 25, 30	0.98
‘20172017’	**MD5:** 29ce822e8b7e51a318e382b0e34f4360	4	3, 4, 8	11, 18, 28,29	1
	Random: Eý÷9ÈSVØ÷Èý({në¦¡»Ëí%o0[°w		1, 2, 5	1, 19, 22 ,23	0.97
‘pass123’	**MD5:** 32250170a0dca92d53ec9624f336ca24	5	1, 2, 6, 8	5, 12, 21, 29, 30	0.98
	Random: ;)Uæ2Ì¥¦Á4Ý&÷¿7õNBàVvÞ		3, 4, 5, 6	5, 6, 11, 17, 30	0.96
'!@#pass123’	**MD5:** 0dbe23f840b9443a25b9b6b49375925b	6	1, 3, 5, 6, 7, 9	5, 16, 19, 25, 28, 31	0.96
	Random: ÅÜpco¥Ä.>»□ÁÒCjk{täëÉc;¢þ&o		3, 4, 5, 8, 10	11, 17, 22, 29, 30	0.96
'123$%^789'	**MD5:** 0f246b38c49c50b7460f82cbe15871e9	7	2, 3, 5, 6, 8, 9	4, 9, 11, 13, 22, 24, 27	0.96
	Random: ³-Ô"+¥□ é:½S$3ÕpnyßᵌⁱDGÑGb		1, 2, 6, 7, 8	11, 14, 15, 17, 30, 31	0.90
'Password123*()'	**MD5:** 35acc07d9d81171d94afff007b2a69fa	8	2, 4, 8, 9, 13, 14	5, 13, 14, 21, 23, 26, 27, 30	0.90
	Random: ò3s?Ð >w¢Bÿ6ªòÙ7ÄËD©»Ê{æË		2, 4, 7, 9, 12, 13	3, 7, 18, 21, 24, 27, 30	0.89
'123Pass#$%^'	**MD5:** d91b9fb6a4c0a5c1d2a8c33b7d8cd261	9	1, 3, 4, 6, 7, 9, 11	5, 8, 9, 16, 17, 18, 20	0.85
	Random: Ü0UÜÍi'¡fêfá Åð'yk÷»»5vu·		3, 4, 5, 6, 8, 10, 11	3, 6, 9, 14, 15, 17, 24, 27	0.83
'P1a@s#s$w%o^r&d'	**MD5**: f0ec1af343ebd56e7aa23c14f99fb7b6	10	4, 8, 9, 13, 15, 23	1, 2, 3, 5, 8, 10, 12 14	0.81
	Random: o9ëAm¶¦À`ÏÅÏ?âï°«ßo±tüéÄ¸5g>ë		4, 9, 10, 11, 12, 14, 15	3, 8, 10, 14, 18, 21, 22, 25	0.81

In some rows of Table 2, the GA obtained less number of positions than dc value, due to dCorr values being optimal. The dc is maximum position counts in each iteration of the algorithm. For example, in Table 2, the row with plaintext equal 'Password123*()' whose dc is 8, in the result of the simulation, the number of positions in plaintext is 6, in hash positions by MD5 is 8, and in the random hash is 7. The difference in the number of positions is due to the maximum dCorr value between plaintext and its hash data that is calculated using the proposed algorithm.

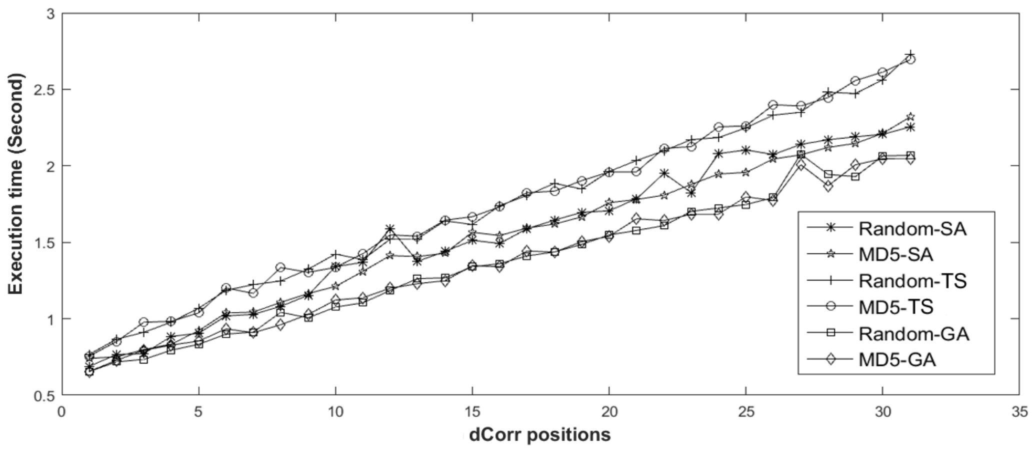

Figure 7. Comparison of Execution Times

Another parameter for the analysis of the proposed method is the execution time. To compare of the proposed algorithm with other methods, TS (Glover & Laguna, 2013) and SA are programmed (Siarry, 2016). The proposed method is first run by GA and the best value of dCorr is calculated before it is run using TS and SA to reach best dCorr value. The plaintexts and their hashes are considered equal for each of three algorithms. Figure 7 shows 32 columns include the execution time (Second) of the proposed algorithm as GA, TS, and SA. In this chart, the horizontal axis as dc is the number of dCorr positions and the vertical axis is execution time of all methods. The executed random hash function has a less time than MD5. In fact, the MD5 method has a longer process than random function. The average of execution time in two methods (MD5, Random) in GA is (1.3596, 1.3647), in TS is (1.7358, 1.72.86), and in SA is (1.5225, 1.5430). As our experiments for three algorithms (GA, TS, and SA), the execution time parameter for GA is minimum and TS has maximum value.

CONCLUSION

In this paper, we prove the existence of a mathematical and logical process for calculating MD5 hash method, which is different with the random function. To do this work, it is very important to find the correlation between the plaintext and its hash in breaking the hash method.

The study has shown different methods are used for correlation measuring between two data vectors. These methods include MIC, dCorr and other correlation measures. The dCorr is the best. We calculate the maximum dCorr value of the best relationship positions in plaintext and its hash by two hash functions as MD5 and random methods. The GA is used for optimal positions. The proposed method is executed for 1000 pairs of plaintexts and their hashes. The results of dCorr values show that MD5 method is better than the random method for many different positions. The execution time of MD5 is also longer than random function due to its long process. This method is proposed with TS and SA algorithms. As results, the average execution time by GA is 22% less than TS and 11% less than SA.

The results of this paper prove a strong relationship between the plaintext and its hash that is computed by dCorr for MD5 and random method. The hash functions with a high dCorr value between plaintext and its hash are not suitable for the message digest and it is simply broken by the attacker. Future research will focus on discovering the relationship in big data by dCorr measurement.

REFERENCES

Budhathoki, K., & Vreeken, J. (2017, June). Correlation by compression. In *Proceedings of the 2017 SIAM International Conference on Data Mining* (pp. 525-533). Society for Industrial and Applied Mathematics.

Choi, J., Park, J., & Kim, H. (2017). Forensic analysis of the backup database file in KakaoTalk messenger. In *Big Data and Smart Computing (BigComp), IEEE International Conference* (pp. 156-161). IEEE.

Choi, Y., Lee, J. Y., Choi, S., Kim, J. H., & Kim, I. (2016). Introduction to a network forensics system for cyber incidents analysis. In *Advanced Communication Technology (ICACT), 2016 18th International Conference on IEEE* (pp. 50-55). IEEE.

De Siqueira Santos, S., Takahashi, D. Y., Nakata, A., & Fujita, A. (2013). A comparative study of statistical methods used to identify dependencies between gene expression signals. *Briefings in Bioinformatics, 15*(6), 906-918.

Downar, L., & Duivesteijn, W. (2017). Exceptionally monotone models—The rank correlation model class for Exceptional Model Mining. *Knowledge and Information Systems, 51*(2), 369-394.

Geerligs, L., & Henson, R. N. (2016). Functional connectivity and structural covariance between regions of interest can be measured more accurately using multivariate distance correlation. *NeuroImage, 135*, 16-31.

Glover, F., & Laguna, M. (2013). Tabu Search*. In *Handbook of Combinatorial Optimization* (pp. 3261-3362). New York, NY: Springer.

Gorgens, E. B., Valbuena, R., & Rodriguez, L. C. E. A. (2017). Method for optimizing height threshold when computing airborne laser scanning metrics. *Photogrammetric Engineering and Remote Sensing, 83*(5), 343-350.

Górecki, T., Mirosław, K., & Woły´nski, W. (2017). Correlation analysis for multivariate functional data. In *Data Science* (pp. 243-258). Springer, Cham.

Heller, R., Heller, Y., Kaufman, S., Brill, B., & Gorfine, M. (2016). Consistent distribution-free k-sample and independence tests for univariate random variables. *The Journal of Machine Learning Research, 17*(1), 978-1031.

Kwok, H. S., & Tang, W. K. (2005). A chaos-based cryptographic hash function for message authentication. *International Journal of Bifurcation and Chaos, 15*(12), 4043-4050.

Linge, Y., Dumas, C., & Lambert-Lacroix, S. (2014). Mutual information coefficient analysis. *International Association for Cryptologic Research, 12*, 2012.

Lipowski, A., & Lipowska, D., (2012). Roulette-wheel selection via stochastic acceptance. *Physica A: Statistical Mechanics and its Applications, 391*(6), 2193-2196.

Martínez-Gómez, E., Richards, M. T., & Richards, D. S. P. (2014). Distance correlation methods for discovering associations in large astrophysical databases. *The Astrophysical Journal, 781*(1), 39-49.

Mitchell, M. (1998). *An introduction to genetic algorithms*. USA: MIT press.

Naboulsi, D., Fiore, M., Ribot, S., & Stanica, R. (2016). Large-scale mobile traffic analysis: A survey. *IEEE Communications Surveys and Tutorials, 18*(1), 124-161.

Pan, W., Wang, X., Wen, C., Styner, M., & Zhu, H. (2017). Conditional local distance correlation for manifold-valued data. In *International Conference on Information Processing in Medical Imaging* (pp. 41-52). Springer, Cham.

Reshef, D., Reshef, Y., Finucane, H., Grossman, S., McVean, G., Turnbaugh, P., … & Sabeti, P. (2011). Detecting novel associations in large datasets. *Science, 334*(662), 1518-1524.

Rivest, R. (1992). The MD5 message-digest algorithm. *MIT Laboratory for Computer Science and RSA Data Security.* Network Working Group.

Schneier, B. (2004, August 19). Cryptanalysis of MD5 and SHA: Time for a New Standard. In *Computer World, Crypto 2004 conference* (pp. 1-2). Santa Barbara, California.

Shao, F., & Li, K. (2016). A graph model for preventing railway accidents based on the maximal information coefficient. *International Journal of Modern Physics B, 31*(3), 1750010.

Siarry, P. (2016). Simulated Annealing. *Metaheuristics. Springer, Cham*, 19-50.

Sun, Y., Kirley, M., & Halgamuge, S. K. (2017). Quantifying variable interactions in continuous optimization problems. *IEEE Transactions on Evolutionary Computation, 21*(2), 249-264.

Székely, G. J., & Rizzo, M. L. (2009). Brownian distance covariance. *The Annals of Applied Statistics, 3*(4), 1236-1265.

Székely, G. J., & Rizzo, M. L., (2013). The distance correlation t-test of independence in high dimension. *Journal of Multivariate Analysis, 117*, 193-213.

Wang, S., Zhao, Y., Shu, Y., & Shi, W. (2017). Improved approximation algorithm for maximal information coefficient. *International Journal of Data Warehousing and Mining (IJDWM), 13*(1), 76-93.

Wang, S., Zhao, Y., Shu, Y., Yuan, H., Geng, J., & Wang, S. (2018). Fast search local extremum for maximal information coefficient (MIC). *Journal of Computational and Applied Mathematics, 327*(1), 372-387.

Wang, T., & Li, Z. (2017). Outlier detection in high-dimensional regression model. *Communications in Statistics-Theory and Methods, 46*(14), 6947-6958.

Wang, T., Zheng, L., Li, Z., & Liu, H. (2017). A robust variable screening method for high-dimensional data. *Journal of Applied Statistics, 44*(10), 1839-1855.

Wang, X., Jiang, B., & Liu, J. S. (2017). Generalized R-squared for Detecting No independence. *Biometrika, 104*(1), 129-139.

Zhai, A., Xu, F., Pan, H., Shi, J., & Xiong, G. (2017). Design for network file forensics system based on approximate matching. In *Advanced Multimedia and Ubiquitous Engineering* (pp. 426-431). Springer, Singapore.

Zhang, Y., & Xiao, D. (2013). Cryptanalysis of S-box-only chaotic image ciphers against chosen plaintext attack. *Nonlinear Dynamics, 72*(4), 751-756.

Zhang, Y., Xiao, D., Wen, W., & Li, M. (2014). Breaking an image encryption algorithm based on hyper-chaotic system with only one round diffusion process. *Nonlinear Dynamics, 76*(3), 1645-1650.