Performance Quantification of the Cryptographic Key Generated using Evolutionary Algorithm

Ragavan M, K Prabu

Abstract: Main heart of the cryptographic process is the key used for encryption and decryption. When the key is very strong and giving a big challenge for breaking it, then, there is an assurance for the high security for the data. Several methods are used for generating the keys. Almost all methods are defined with high randomness in key generation, which gives the assurance for difficulties in key detection. In this paper, the evolutionary method, genetic algorithm is used for key generation. The algorithm is framed with the population of gene size as 16 bytes. Initial population with M size is generated and the best genes are selected by applying the fitness function. From the selected genes, the crossover and mutation process are applied. From the outcome of the operations, the best fitted genes are added into the population. This process will be done repeatedly until 'n' times. From the final population set, the genes are randomly selected for the key formation. For 16 byte key, a single gene is considered. For a 32 byte size key, two genes are randomly selected and merged to generate the required key for cryptographic process. The performance of the key generated by evolutionary method is compared with the two different built-in key generation methods such as random key generation method and password key generation method in Python. Performance is evaluated based on the processing time, CPU temperature, entropy values and encrypted data size. The performance of the Evolutionary key generation (EKG) method is almost equal to Password key generation (PKG) method and EKG is looking much better than Random key generation (RKG) method.

Keywords: Genetic Algorithm, Encryption, Decryption, Performance Evaluation, Key Generation.

I. INTRODUCTION

In this digital era, the volume of data generation is significantly high. Starting from the simple and static data such as examination result published in online to the sensitive and highly dynamic data such as the bed-ridden patients health updates over cloud using IoT. All type of data needs high security. Data security can be provided by two ways, symmetric and asymmetric cryptographic process [1]. In symmetric process, both encryption and decryption of the data is done by one key. In asymmetric process, encryption of the data is done by one key, which is referred as private key and the decryption of the data is done by another key, referred as public key.

II. RELATED WORK

Symmetric key algorithms such as DES, 3DES, AES and Blowfish are compared on rounds, block size, key size, encryption/decryption time and CPU process time as parameters. Blowfish algorithm is performed better than all other algorithms. The Blowfish algorithm has performed better with less processing time, low power consumption with high throughput [2].

Cipher MSC-IM is designed based on MSC-1. In MSC-1M, the supporting block length is 256 and the key length considered for the cryptographic process is 256 and 512 bits. The encryption is done with cyclic transformation with encryption cycles as 9 and randomization at the end [3].

Two symmetric algorithms such as AES and DES are compared by the performance evaluation factors such as the speed, time and cost. Visual Studio with VC++ is used for collecting the performance factors and .Net along with the word crypto-tool are used for crypto process [4].

An improved symmetric type of algorithm, Blowfish algorithm is designed for providing high security for the multimedia files which are stored and transmission over internet. It is designed in such a way to provide security for text, audio and video forms of data. The performance of the algorithm is compared with other symmetric algorithms such as AES, DES and Triple DES. The encryption and decryption times are considered as the comparison parameter to evaluate the improved Blowfish algorithm. [5]

Audio data is encrypted with two phases. Random number generator based on the cellular automata (CA) is the first phase and data encryption of the audio file using RC5 is the second phase. Rule scheduling algorithm is used to select the rules set to be applied to generate the sub keys based on CA. Two words length of the block size, which is equal to eight bytes, is used in the enhanced RC5. Each block of data is encrypted by different sub keys. This is mainly applied to strengthen the security of the encrypted data [6].

III. CRYPTOGRAPHIC PROCESS

The cryptographic process is done in two phases: first one, key generation using evolutionary method (Genetic algorithm) and the second phase is about to evaluate the performance of the genetically generated key along with random key generation method and password key generation method.
A. Evolutionary Algorithm for Key Generation

The Genetic algorithm based key is referred as Evolutionary Key Generation (EKG) is generated with six steps: initial population (gene) generation, best gene selection based on the fitness function, crossover operation, mutation operation, best gene selection for final population and key extraction from the final population set. Key generation process is shown in Fig 1.

Initial Population

Each gene is defined with 16 bytes length. Totally M genes are generated. Finally, the initial population is with size Mx16 genes.

Best fitted initial Population

From the initially generated population gene set, only the best genes are required to select for the next process. Fitness function in equation (1) is used for selecting the best genes from the initial population set. The fitness values of the chromosomes > .5 are selected as best fitted population.

\[
\text{Best}_{\text{Fit}} = \frac{\sum_{i=1}^{M} \frac{\text{Gene}_{i}}{N}}{M} \quad (1)
\]

where,
- M – Number of Chromosomes (Population size)
- N – Number of Genes in the Chromosomes (16 bytes)
- Gene[i] – ASCII value of j\text{th} gene in i\text{th} chromosome
- Best_fit – a fractional value ranging from 0 to 1

Fig. 1. Key Generation Process using Genetic Algorithm

Crossover operation

Crossover operation is performed to swap the selected number of genes between selected chromosomes. To make the operation, the following algorithm is applied to perform the crossover operation

Crossover Algorithm

For all chromosomes in the init_best_fit_population

\[ \text{p1} \leftarrow \text{select a random value between 0 to .9} \]

If (p1 > .5)

then

\[ \text{Generate 2 random integers r1 and r2 between 0 to m-1.} \]

[Chromosomes for crossover]

Generate a random integer r3 between 0 to n/2.

[Number of genes to be crossed over]

For gene count in r3

\[ \text{Generate 2 random integers r4 and r5 between 0 to n-1.} \]

[Gene position to be swapped]

Swap the genes gene(r1,r4) and gene(r2,r5)

else

Update the crossover_genes

Continue the outer ‘for’ loop.

Mutation Operation

Next important genetic operation applied on the chromosome is Mutation. It is applied on the chromosomes to make genetic difference from the parent chromosomes to the next generation chromosomes [7]. The mutation may generate a new off spring which totally different from its parent. In this paper, two different types of mutations are applied, left shift / right shift Mutations. The Mutation algorithm is given below.

Mutation Algorithm

\[ \text{op} \leftarrow \text{generate a random integer between 1 and 2} \]

if op = 1

then

\[ \text{do left_mutation(CrossoverGenes)} \]

else if op = 2

then

\[ \text{do right_mutation(CrossoverGenes)} \]

left_mutation(CrossoverGenes)

for every chromosome in CrossoverGenes

Rotation_value \leftarrow \text{generate a random number between 0 to 15} \]

[Number of genes to be rotated]

Make the left shift of ‘Rotation_value’ genes of the particular chromosome

Update the mutation_genes

right_mutation(CrossoverGenes)

for every chromosome in CrossoverGenes

Rotation_value \leftarrow \text{generate a random number between 0 to 15} \]

[Number of genes to be rotated]

Make the right shift of ‘Rotation_value’ genes of the particular chromosome

Update the mutation_genes

Final Population set

Best fitted chromosomes are extracted from the mutation_genes set and added into the best fit population set, which is referred as final population set.
Key Extraction

From the final population set, the chromosomes are randomly picked to frame a key. One chromosome is framed with 16 bytes length. If a key is 32 bytes, then, two non-similar chromosomes are randomly picked to frame the key. In this paper, the key size is fixed as 32 bytes.

B. Performance Evaluation of the Cryptographic Keys

The second phase of the cryptographic process in this paper is to measure the performance of the key which is generated using Genetic algorithm with other keys. In Python, two different ways are applied for key generation. They are, random key generation (RKG) and password key generation (PKG). For using these two methods two different modules are needed to be installed. Table 1 shows the modules which are required for the key generation.

<table>
<thead>
<tr>
<th>Description</th>
<th>Key Generation Techniques</th>
</tr>
</thead>
<tbody>
<tr>
<td>Python Module</td>
<td>Fernet</td>
</tr>
<tr>
<td>Random Key Generation</td>
<td>secrets and encrypt</td>
</tr>
<tr>
<td>Fernet.generate_key()</td>
<td>script.hash()</td>
</tr>
<tr>
<td>Key Size</td>
<td>32 bytes</td>
</tr>
</tbody>
</table>

Table 1. Supporting tools for Key generation in Python

The performance of the three types of keys can be evaluated through some cryptographic algorithms. Here, AES (Advanced Encryption Standard) algorithm is considered for the performance analysis of the keys. A word document file with size 342 KB is considered as an input .doc document for encryption and decryption process. AES encryption and decryption process is shown in Fig. 2.

Fig. 2. Cryptographic Process with Three different keys

The factors such as the processing time, the CPU temperature, entropy values measured for the encrypted data and the data size are used in this paper. The processing time is calculated using ‘time.clock( )’ method. This is applied before and after the encryption/decryption process. The difference between the two clock( ) function is representing the processing time. CPU temperature is measured using the module ‘wmi’. Windows Management Instrumentation (WMI) is used to provide any information about a computer system [8]. Along with this, ‘OpenHardwareMonitor’ package is also required to install in the system. The execution of this command results the number of cores, packages and CPU temperature. The entropy value for the encrypted message is measured by applying three methods. The highest entropy value is assuring the strong security of the message. Trying to extract the plain text from the cipher text is really difficult for the cipher text with high entropy. Finally, the original and encrypted data length is measured by using byte counts.

IV. RESULTS AND DISCUSSION

AES encryption/decryption for the plain text is done separately with the three different keys. The performance evaluating factors are evaluated separately. The analysis of the processing time is shown in Fig 3. In the Figure, ‘Enc-Time’ specifies the encryption time and ‘Dec-Time’ specifies the decryption time. The processing time taken by Random Key Generation (RKG) is higher than other two methods. Time taken by the Evolutionary Key Generation (EKG) method is higher than Password Key Generation (PKG) method, but lesser than RKG method. The processing time is measured in milliseconds.

The CPU details are also extracted for comparing the three methods of crypto process. Fig 4 shows the CPU temperature evaluation along with some other additional features. System in which the software is developed is a dual core processor. This makes to produce the two core processors core 1 and core 2 information,. Additionally, the packages details are also shown along with the CPU temperature.
The core 1, core 2 and packages count of RKG and PKG are looking less than 50 but, the EKG shows above 50. This is ensuring that, the first two are done with built-in functions but, the third method EKG is designed to complete the Phase I process. This makes the core and packages more than other two methods. In all three methods, the CPU temperature is same as $38^\circ$.

Next, the entropy values of all the three methods are evaluated using three entropy methods such as natural, Shannon and Hartley. This comparison is shown in Fig 5. Among the three entropy evaluation methods, the Shannon method has produced the highest entropy value in all the three cryptographic algorithms. The EKG entropy value is same as PKG entropy values, but, RKG entropy values are looking lesser than the other two methods.

From the above four factors, it is clear that, the cryptographic process of Evolution based Key (Genetic Key) is performing almost similar as PKG method. The evolved factors are revealing the performance of EKG is better than RKG but, almost similar as PKG method.

V. CONCLUSION

The cryptographic process is done with symmetric key based AES algorithm. The key is playing a vital role in the process. A key is generated using Evolutionary algorithm (Genetic Algorithm), it referred as EKG (Evolutionary Key Generation). The performance of the generated key is assessed with other two methods of keys generation techniques such as Random Key Generation (RKG) and Password Key Generation (PKG) using Python.
Three keys are applied on AES algorithm separately and a plain .doc document is considered as input. Four performance evaluation factors such as the processing time, the CPU temperature, entropy values and data length are considered to measure the strength of the keys. The performance of EKG method is almost equal with PKG method and better than RKG method.

REFERENCES

8. https://pypi.org/project/WMI/1.0/, as on 26/12/2019.

AUTHORS PROFILE

Mr. M. Ragavan received his M.C.A from Alagappa University, Karaikudi and M.Phil from Bharathidasan University, Tiruchirappalli, India. Now doing his Ph.D research in Sudharsan College of Arts & Science, Pudukkottai, Tamilnadu, India. His Research Interests are Wireless Sensor Networks (WSNs), Network Security. He has published more than 3 technical papers at various National / International Conferences and Journals.

Dr. K. Prabu received his MCA and M.Phil from Annamalai University, Chidambaram, India. He received his Ph.D Degree in Computer Applications from Manonmaniam Sundaranar University, Tirunelveli, India. He is now working as an Associate Professor in PG & Research Department of Computer Science, Sudharsan College of Arts & Science, Pudukkottai, Tamilnadu, India. He is a Reviewer of 06 National/International Journals. His Research Interests are Adhoc Networks, Wireless Networks & Mobile Computing, and Wireless Sensor Networks. He has published more than 75 technical papers at various National / International Conferences and Journals. He is a life member of ISTE, IACSIT, IAENG, and also senior member of IASED, and IRED.