DOI: http://dx.doi.org/10.26483/ijarcs.v8i7.4397

ISSN No. 0976-5697

Volume 8, No. 7, July – August 2017

International Journal of Advanced Research in Computer Science

REVIEW ARTICLE

Available Online at www.ijarcs.info

RANDOMIZED DES USING IRREDUCIBLE POLYNOMIAL OVER GALOIS FIELD GF(7^3)

J K M Sadique Uz Zaman
Department of Computer Science
A.P.C. Roy Govt. College
University of North Bengal, Siliguri, India

Ranjan Ghosh
Department of Radio Physics and Electronics
University of Calcutta
Kolkata, India

Abstract: Multiplicative inverses under an irreducible polynomial over Galois Field GF(p^m) play important role in cryptography. The substitution box of Advanced Encryption Standard is designed in 1999 with multiplicative inverses under the first irreducible polynomial over GF(2^8). In the present paper, a recently published pseudorandom number generator GF7 designed over GF(7^3) in 2016, is intuitively incorporated into the Data Encryption Standard algorithm to modify it by increasing its randomness as well as security. Following the method proposed by National Institute of Standards and Technology, the statistical randomness testing is done on a large number of output ciphers obtained by the algorithm. The statistical result shows tremendous improvement in randomness for the proposed modified algorithm.

Keywords: Galois field, GF(7^3), Irreducible polynomial, Multiplicative inverse, Randomness, DES, Symmetric cipher.

I. INTRODUCTION

The DES, an acronym representing Data Encryption Standard, is an important private key symmetric block ciphering algorithm introduced in 1977 by the National Bureau of Standards, USA [1], [2] and is still considered to be an important algorithm by many researchers [3], [4], [5]. Here the aforesaid algorithm is chosen to study the effect of randomness by incorporating a suitable Pseudo Random Number Generator (PRNG) into it. In the chosen algorithm; 8-character plaintext is encrypted with 8-character key using five P-boxes, eight 4-bit S-boxes, key-bits shifting method and an expansion algorithm – all are of fixed types. The same 8-character key is repeatedly used to encrypt many 8-character text blocks comprising a long message. The 8th bit (LSB) of each of the 8 key characters are dropped thereby making the key effectively of 56 bits instead of 64 bits. Since its inception it was considered to be a very robust algorithm till 1994 when its key could be made known using a “DES Cracker” machine to decode a message within 3 to 4 hours of its receipt [6]. It may be noted that use of same key for each and every block is a serious weakness including the fact that the key is short too. These weaknesses can be overcome if randomization technique based on an appropriate PRNG is suitably introduced.

All elements in DES algorithm fit particularly well together and changing the slightest little element in it is more likely to introduce weaknesses rather than to strengthen the algorithm. Yet an initiative has been taken to improve its security. The PRNG would render more immunity to this algorithm and make it more competitive with respect to other ciphering algorithms. The purpose of the present paper is to study the effect of increasing the key length and of incorporating a PRNG in the chosen algorithm. A number of good PRNG is available in literature [7]. And most recently a new PRNG has been developed and published in 2016 using the multiplicative inverses under the first irreducible polynomial over GF(7^3) [8]. This PRNG is innovatively incorporated in the chosen algorithm; the output is statistically tested by NIST methodology and observed a marvelous improvement in randomness [9], [10], [11]. The NIST test is executed on three algorithms DES56, DES64 and RDES – where for each and every algorithm 300 ciphered bit files each on consisting 1344000 bits are generated using 300 different keys. The fifteen tests proposed by the NIST are applied on such 900 bit files and the results obtained are presented and discussed later in this paper. It is expected that through the present study; scopes to introduce randomness using suitable PRNG in other cryptographic algorithms would also be visible.

A brief overview of the DES algorithm is presented in Sec. II and two attempts to increase its key size are discussed in Sec. III. As mentioned above three different softwares are developed based on three algorithms and a large number of data is generated for statistical randomness testing. The testing process is presented in Sec. IV and the results obtained by the statistical tests are discussed in Sec. V. The conclusion of the study is drawn in Sec. VI.

II. OVERVIEW OF THE DES ALGORITHM

All the operations in the DES algorithm are executed in bit level. The nonlinearity approach of Boolean function is considered to design the S-boxes of this algorithm and that’s why these S-boxes are appropriate. The size of key is 56 bits long that were chosen from the 8 key characters dropping the 8th bit of each character. In literature this is mentioned as dropping of parity bit [12], [13]. Considering the DES algorithm and following all the permutation processes related to text-bits and key-bits, expansion of text bits, circular shifting of key-bits and the substitution processes using eight S-boxes as mentioned in standard literatures [2], [12], [13], a program code DES56 has been developed by dropping the 8th bit of all the key characters. Here the program code is so named since 56 bits are taken from a given 8-character fixed key. It has been found that there are (2^8 – 1) duplicate keys for an eight byte text block which is ciphered with an 8 byte key.

The concept of dropping was justified during mid-seventies of twentieth century when the ASCII was of 7 bits and the 8th bit was considered as LSB and used to indicate the parity bit of the binary coded character [14]. In those days the ASCII was for 128 characters. Presently there is no parity bit concept in ASCII and each character is now of 8 bits making an accommodation of 256 characters instead of 128 [15]. Thus in today's perspective there is no sense of dropping the 8th bit of each character.

For a certain cipher message if all the 8 key characters are known and for complete message recovery if one endeavors to search their particular arrangement by applying brute force method, the DES decryption code has to be run factorial 8 (8!) times or less. In the event the key is composed of any eight alphabetic characters including the uppercase and lowercase ones, for searching the key the total number of times the DES decryption code is to be run is equal to 52^8 or less. If all the 8 key characters do belong to alphanumeric ones, the above mentioned computational effort would be 62^8 or less. If it is assumed that the key can be composed of any eight characters among all the available alphanumeric and special characters, the computational effort will be $256^8 = 2^{64}$ times or less. The DES could survive from attacks during the first 17 years of its inception, since such a gigantic computational effort was humanly impossible. One can note that 56 bit key is too short a key for a dedicated fast hardware [16]. In fact, the programming support to the brute force key searching activity using fast hardware had been possible because of the fixed nature of the key.

III. INCREASING THE KEY SIZE AND RANDOMNESS

It is evident from the above discussion that the cracking becomes a difficult job if key size is increased and random property is incorporated to the fixed nature of the key. In this paper two approaches are considered to enhance the security of DES, one is to increase the key size from 56 bits to 64 which is discussed in Sec. III.A, and another is to incorporate an appropriate PRNG at exact position of the algorithm, that is presented in Sec. III.B.

A. *Key length 64 bit: Not better than 56*

If all the 64 key-bits are to be considered, one has to incur some changes on three issues in DES56 in order to develop a new program DES64 - the program code is so named since all the 64 bits are considered from a given 8-character key. These three issues are (i) different permutation guide for 64 key-bits instead of 56 key-bits, (ii) different circular shifting algorithm of both the key halves consisting of 32 bits instead of 28 bits and (iii) different permutation guide to choose 48 bits from the shifted and then concatenated two key-halves of 64 bits instead of 56 bits.

The actual changes that are undertaken in DES56 are related to the initial processing of key where 64 key bits are used. For circular shifting of key bits in each round, necessary changes are made in algorithm to accommodate 32 bits in each key halves. A modification is undertaken also in permutation guide to choose 48 bits from 64 of two shifted and concatenated key-halves. The other three P-boxes, the eight S-boxes and the expansion algorithm used in DES56 remain the same as mentioned in literatures. It is observed that the outcome of DES64, presented in Sec. 5, is not better than DES56. Hence in next step a PRNG is applied only in DES56 not in DES64.

B. *Incorporating a suitable PRNG*

The authors have published recently a new PRNG GF7 [8]. In this paper the GF7 is innovatively incorporated into the DES algorithm and a new program Random DES (RDES) is developed [7]. The output cipher obtained by the proposed algorithm are tested statistically and observed that the randomness is tremendously improved. Here the new PRNG GF7 is narrated below:

In GF7 random shuffling is used on the multiplicative inverses under the first irreducible polynomial $x^3 + 2$ over GF(7^3). The zero has no inverse and in GF(7^3) there are 342 inverses. The first position of an linear array say G[256] is filled by 0 and the next 255 elements are filled by the multiplicative inverses whose value is less than $(256)_{10}$ and the rest 87 inverses are complemented and stored in the first 87 positions of another linear array K[256]. The next 169 positions of the K array are filled recursively by the user given key characters. In present paper, all the 300 different keys are 8 characters long and same as used in DES56 and DES64, though RDES can accommodate a long key up to 177 characters. The design procedure of GF7 is given below:

```
Initialization of G[256] array
G[0]=0;
for i = 1 to 255 do
    G[i] = Sequentially filled up by the multiplicative
           inverse over GF(7^3) which is less than (256)_10.

Initialization of K[256] array
for i = 0 to 86 do
    K[i] = Sequentially filled up by the complement of
           multiplicative inverse over GF(7^3) which is
           greater than (255)_10.
for i = 87 to 255 do
    K[i] = Recursively filled up by the user given key
           characters.

Permutation of G[256] array
j = 0;
for i = 0 to 255 do
    j = (j + G[i] + K[i]) mod 256;
    Swap (G[i], G[j]);
```

After permutation of G array it will produce a series of random byte R as follow:

```
i = j = 0;
while (true)
    i = (i + 1) mod 256;
    j = (j + G[i]) mod 256;
    Swap (G[i], G[j]);
    t = (G[i] + G[j]) mod 256;
    R = G[t];
```

The above discussed GF7 PRNG is added in DES56 encryption algorithm after execution of final permutation at the end of 16th round. The Data Flow Diagram of the same is given in ANNEXURE-1 where it is clearly shown that the 64 output bits are exclusive-ORed with the 64 random bits at the lower left corner of the thick lined box. The same procedure continues for next blocks till the end of the message. The Data Flow Diagram for corresponding decryption algorithm is given in ANNEXURE-2. From the diagram one can easily understand where and how the PRNG should be added. It is evident that 64 random bits will be exclusive-ORed with the 64 cipher bits before the initial permutation which is shown at the upper left corner of the thick lined box.

The PRNG produces a random byte in each and every execution; hence the concept of fixed key in DES56 is discarded. On the other hand, since the GF7 can accommodate a key of length 169 bytes; the concept of 8 characters key is also removed. The RDES can take any length of key between 8 and 177 (8 + 169 in GF7) characters while it is fixed to 8 characters in DES56.

IV. OVERVIEW OF METHODOLOGY FOR STATISTICAL TESTING

The fifteen statistical tests proposed by NIST are applied on a set of long sequences of bits (approximately 1.34 million bits) to check randomness of the bit sequences thereby the algorithm that produces the set of bit sequences. In this paper testing is done to check the random property of the proposed algorithm RDES by comparing its result with the same of DES56. The computational procedures are presented in NIST statistical testing documents [10], [11] and a review work is also available in [9]. The documents explain very nicely the calculation techniques of Probability value (P-value) where one can see that the P-value is calculated using the χ^2-value coupled with the degrees of freedom and not only based on the χ^2-distribution function. The passing criterion for a particular test the P-value is set as; P-value ≥ 0.01 (where 0.01 is considered as significance level α). Based on the P-values obtained from all the tested bit sequences for a particular test, NIST suggest a statistical method to calculate proportion of passing. The distribution pattern of P-values is also checked to see the uniformity distribution. The minimum length of bit sequence required for various tests is different and information for this is available in [9].

Here 300 different 8-character keys are used to encrypt a message of length 168000 bytes for the three algorithms DES56, DES64 and RDES; and each algorithm generates 300 different bit-sequences each of 1344000 bits long. The frequency distribution of P-values obtained for fifteen tests of these three algorithms are shown in Table I, Table II and Table III respectively.

Table I. Frequency distribution of P-values of DES56

Test No.	*0.0-0.01*	*0.01-0.1*	*0.1-0.2*	*0.2-0.3*	*0.3-0.4*	*0.4-0.5*	*0.5-0.6*	*0.6-0.7*	*0.7-0.8*	*0.8-0.9*	*0.9-1.0*
1	15	54	40	34	33	24	15	26	22	14	23
2	7	47	32	22	31	26	20	21	27	18	49
3	15	48	45	29	25	28	24	20	18	18	30
4	24	55	42	34	29	30	24	16	19	17	10
5	88	117	39	11	12	9	5	6	6	5	2
6	4	26	32	24	24	44	17	43	37	22	27
7	11	43	39	29	26	31	28	27	24	23	19
8	16	59	42	29	28	20	24	24	21	19	18
9	6	36	34	27	33	27	18	22	27	36	34
10	3	31	27	30	31	26	39	27	29	27	30
11	72	152	85	57	58	42	39	23	30	23	19
12	37	82	54	32	19	19	19	13	10	10	5
13	9	43	51	66	58	58	51	64	61	69	70
14	36	203	220	235	249	257	255	230	252	235	228
15	55	415	557	553	568	538	540	551	551	525	547

Table II. Frequency distribution of P-values of DES64

Test No.	*0.0-0.01*	*0.01-0.1*	*0.1-0.2*	*0.2-0.3*	*0.3-0.4*	*0.4-0.5*	*0.5-0.6*	*0.6-0.7*	*0.7-0.8*	*0.8-0.9*	*0.9-1.0*
1	22	52	34	33	24	23	27	21	15	24	25
2	11	31	33	31	25	18	24	28	28	31	40
3	21	48	31	48	25	24	20	15	25	28	15
4	17	57	49	31	25	25	19	26	23	16	12
5	90	112	35	19	22	6	7	3	3	1	2
6	2	20	38	23	22	45	18	41	33	28	30
7	8	50	37	37	29	20	27	21	29	23	19
8	14	65	40	24	32	29	26	24	13	17	16
9	7	45	27	32	35	14	25	28	28	29	30
10	3	29	32	29	17	34	41	26	30	33	26
11	66	138	88	74	48	48	31	31	30	29	17
12	41	73	45	34	21	28	14	17	13	9	5
13	6	58	62	63	70	58	66	50	61	58	48
14	32	213	218	234	244	202	259	244	239	264	251
15	65	444	523	537	525	522	557	539	573	540	575

Two important parameters (i) Threshold value (T-value) and (ii) P-value Of P-values (POP) are considered for measuring the degree of randomness of the output bit sequences obtained from an algorithm. The mathematical concept behind the T-vale and POP are briefly explained in Sec. IV.A and in Sec. IV.B respectively. The statistical results of the three algorithms DES56, DES64 and RDES are presented and discussed in Sec. V.

Table III. Frequency distribution of P-values of RDES

Test No.	*0.0-0.01*	*0.01-0.1*	*0.1-0.2*	*0.2-0.3*	*0.3-0.4*	*0.4-0.5*	*0.5-0.6*	*0.6-0.7*	*0.7-0.8*	*0.8-0.9*	*0.9-1.0*
1	5	26	23	32	38	26	25	28	25	36	36
2	4	35	32	36	35	22	30	29	17	31	29
3	2	27	29	22	36	31	27	36	36	23	31
4	2	30	39	26	30	33	34	30	27	28	21
5	4	29	21	25	28	31	28	35	31	30	38
6	8	27	34	37	22	40	24	30	30	23	25
7	4	25	30	27	33	27	29	28	31	30	36
8	4	32	30	35	29	28	22	32	28	22	38
9	9	18	31	29	23	24	25	40	33	36	32
10	2	25	32	24	20	29	29	25	25	52	37
11	9	43	57	64	39	64	68	74	62	57	63
12	4	22	31	28	25	30	23	45	30	30	32
13	2	65	55	60	52	59	51	57	64	62	73
14	30	208	240	216	224	222	258	266	250	237	249
15	68	441	502	516	585	556	548	561	557	517	549

A. T-value calculation: Observed Proportion Of Passing (OPOP)

It is necessary to have a large number of samples of bit sequences produced by an algorithm to estimate the Observed Proportion Of Passing (OPOP) of a particular test. If n samples of bit sequences are tested by a test which produces one P-value, then the statistical average of T-value would be,

$$T_{value} = (1-\alpha) - 3\sqrt{\frac{\alpha(1-\alpha)}{n}} \qquad (1)$$

Here the significance level α = 0.01. The size of n should be greater than the inverse of α. The T-value = 0.972766 for n = 300. This means that such a test is to be considered as statistically successful, if at least 292 P-values out of the 300 do pass the test. If any test produced r number of P-values, then for calculating the T-value in equation (1), one should consider (n×r) instead of n. If the significance level α and sample size n remains same, then the T-value is 0.983907 for r = 8 (test number 14 in Table IV). Such a test is considered statistically successful if at least 2362 P-values out of the total 2400 do pass the test. If the OPOP value is greater than or equal to the corresponding T-value then the status for proportion of passing a particular test would be considered as success.

B. *POP calculation: Distribution pattern of P-values*

One can understand the distribution pattern of P-values obtained from a large number of bit sequences (sample size) for a particular test; here the sample size is taken as 300. The P-values may or may not be distributed uniformly throughout the region between 0 and 1. The P-values for a particular test are classified in 11 sub-intervals between 0 and 1 in Table I, Table II and Table III.

To estimate the χ^2-deviation of distribution of P-values, the range of P-value (0 – 1) is classified in 10 groups. The first two groups (0.0 – 0.01) and (0.01 – 0.1) of P-values are merged together as one group and the rest in 9 groups. The χ^2-deviation of distribution of P-values is computed as,

$$\chi^2 = \sum_{i=1}^{10} \frac{\left(M_i - \frac{n}{10}\right)^2}{n/10} \qquad (2)$$

where, M_i is the number of P-values in a group i, and n is the sample size. If a particular test produces r number of P-values, then n = r × sample size = r×n. In this testing, degrees of freedom $\nu = 9$ and the two parameters x and y in the gamma function $\Gamma(x, y)$ are taken as,

$$x = \nu / 2 \quad \text{and} \quad y = \chi^2 / 2$$

and the corresponding POP is calculated as,

$$POP = 1 - \frac{\Gamma(x, y)}{\Gamma(x, \infty)} \qquad (3)$$

The P-values are considered to be uniformly distributed if POP ≥ 0.0001 in eq.(3).

V. Results: Comparative Study of DES56, DES64 and RDES

To analyze the data obtained from statistical testing the counting of P-values are given in Table I, Table II and Table III respectively for DES56, DES64 and RDES algorithm. The P-value data are divided into 11 groups between 0 and 1. Depending on the value of P-value for a particular test, the count of an appropriate group is increased in which the particular P-value belongs. For a particular test, a P-value that is considered as unsuccessful (P-value < 0.01) will be entered in column 1of respective Table. Let, S_{10} be the sum of last ten columns and S_{11} be the sum of eleven columns, then the OPOP is the value of S_{10}/S_{11}. It is compared with the T-value and if OPOP ≥ T-value then the bit sequence will be considered as random for that particular test. The POP is also calculated from Table I, Table II and Table III using eq.(3). If POP≥ 0.0001 the distribution of P -values will be considered as uniform.

Following the aforesaid procedure, the test-wise OPOP and POP data for DES56, DES64 and RDES algorithms are calculated for all the fifteen tests. The OPOP data along with the T-value are shown in Table IV where the marking ‘Y’ indicates “Passed” and ‘N’ indicates “Not Passed”. It has been observed that the RDES has passed 14 tests while both the DES56 and DES64 have passed only 7 tests.

The POP results for the three algorithms DES56, DES64 and RDES are presented in Table V where the marking ‘Y’ indicates “Uniform distribution of P-values” and ‘N’ indicates “Non-uniform distribution of P-values”. It has been observed that in both the DES56 and DES64 the P-values are uniformly distributed only for 7 tests and in RDES the P-values are uniformly distributed for all the 15 tests. Regarding the uniformity of distribution of P-values, one can correlate the POP value shown in Table V with a corresponding histogram obtained from the right data of Table I, Table II and Table III. It is evident from Table V that, in RDES the POP value is in the order of 10^{-1} for 14 tests while it is only for 4 tests in both the DES56 and DES64. This indicates that the RDES algorithm produces more uniform data than other two algorithms.

Table IV. Observed Proportion Of Passing (OPOP) for DES56, DES64 and RDES

Test No.	***T-value***	***Observed Proportion Of Passing (OPOP)***		
		DES56	***DES64***	***RDES***
1	0.972766	0.950000 N	0.926667 N	0.983333 Y
2	0.972766	0.976667 Y	0.963333 N	0.986667 Y
3	0.972766	0.950000 N	0.930000 N	0.993333 Y
4	0.972766	0.920000 N	0.943333 N	0.993333 Y
5	0.972766	0.706667 N	0.700000 N	0.986667 Y
6	0.972766	0.986667 Y	0.993333 Y	0.973333 Y
7	0.972766	0.963333 N	0.973333 Y	0.986667 Y
8	0.972766	0.946667 N	0.953333 N	0.986667 Y
9	0.972766	0.980000 Y	0.976667 Y	0.970000 N
10	0.972766	0.990000 Y	0.990000 Y	0.993333 Y
11	0.977814	0.880000 N	0.890000 N	0.985000 Y
12	0.972766	0.876667 N	0.863333 N	0.986667 Y
13	0.977814	0.985000 Y	0.990000 Y	0.996667 Y
14	0.983907	0.985000 Y	0.986667 Y	0.987500 Y
15	0.985938	0.989815 Y	0.987963 Y	0.987407 Y

Table V. P-value Of P-values (POP) for DES56, DES64 and RDES

Test No.	***P-value Of P-values (POP)***		
	DES56	***DES64***	***RDES***
1	8.357485e-13 N	1.155444e-13 N	4.685950e-01 Y
2	8.473000e-07 N	7.571925e-02 Y	1.750485e-01 Y
3	2.151013e-09 N	4.346517e-14 N	5.544205e-01 Y
4	8.252308e-21 N	6.800961e-18 N	6.024576e-01 Y
5	1.411665e-245 N	4.008232e-238 N	6.232397e-01 Y
6	3.711763e-03 Y	3.200570e-03 Y	2.058966e-01 Y
7	4.744493e-04 Y	4.112612e-06 N	9.850350e-01 Y
8	1.841505e-15 N	3.312349e-19 N	4.496721e-01 Y
9	8.558677e-02 Y	1.363687e-03 Y	4.372742e-01 Y
10	8.569072e-01 Y	2.209308e-01 Y	3.621408e-03 Y
11	6.887581e-115 N	1.112691e-92 N	1.296197e-01 Y
12	1.732921e-70 N	1.866477e-60 N	2.896675e-01 Y
13	5.544204e-01 Y	6.405900e-01 Y	6.405900e-01 Y
14	7.424178e-01 Y	1.702939e-01 Y	3.676129e-01 Y
15	1.949687e-01 Y	5.170725e-01 Y	1.945802e-01 Y

From Table V one can observe that, for the DES56 algorithm test number 10 is seen as the best POP obtained from data of test number 10 in Table I – and the same data is displayed below in a corresponding histogram in Fig. 1.

From the histogram it is observed that, the uniformity of P-value distribution is visually evident.

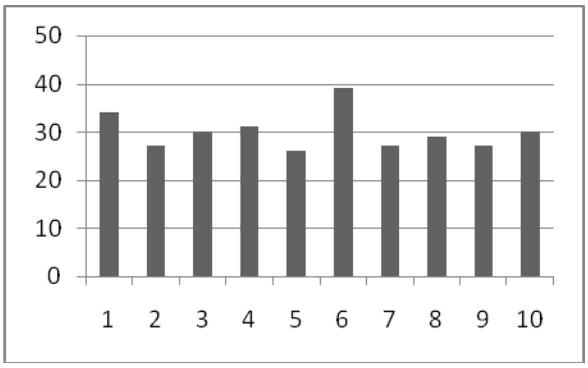


Fig. 1. Histogram for Test no. 10 of DES56
(POP: 8.569072e-01)

The histogram for worst uniformity of DES56 related to test number 5 is given in Fig. 2 where the non-uniformity of P-value distribution is visually realized. For the RDES algorithm test number 7 is seen as the best POP obtained from data of test number 7 in Table III– the same data is displayed in a corresponding histogram in Fig. 3 where the uniformity of P-value distribution is visually evident. The histogram for the worst uniformity of RDES is related to test number 10 and is given in Fig. 4 that also indicates the uniformity of P-value distribution. In all the presented histograms, there are ten columns: first column indicates the number of P-values lying between 0.0 and 0.1; second column indicates the number of P-values lying between 0.1 and 0.2, so on and so forth.

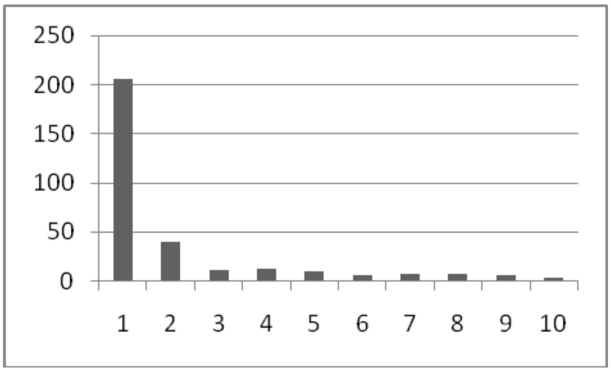


Fig. 2. Histogram for Test no. 5 of DES56
(POP: 1.411665e-245)

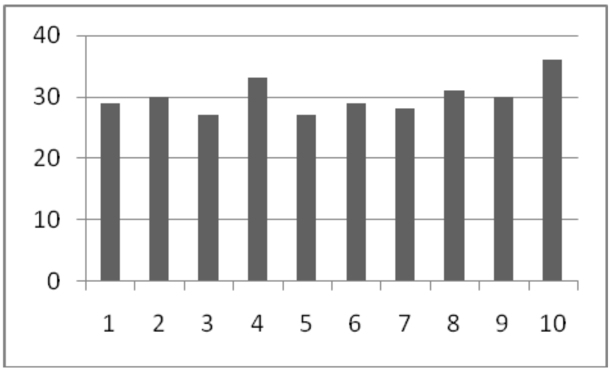


Fig. 3. Histogram for Test no. 7 of RDES
(POP: 9.850350e-01)

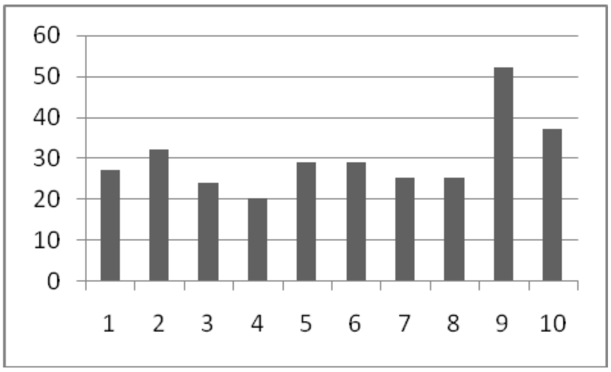


Fig. 4. Histogram for Test no. 10 of RDES
(POP: 3.621408e-03)

VI. CONCLUSION

Incorporating the multiplicative inverses under the first irreducible polynomial over GF(7^3), a randomized DES ciphering algorithm RDES is developed which produces more random cipher as output in comparison to the existing DES cryptosystem. The randomization is introduced successfully in RDES through a PRNG GF7 whose calculation depends on the multiplicative inverses over GF(7^3). As the GF7 can accommodate a key of length 169 bytes, that's why RDES can take any length of key between 8 and 177 bytes. The RDES shows marvelous improvement in statistical randomness due to the GF7 because it produces a random byte in each execution. From statistical result, it is to be concluded that the RDES is a very robust algorithm and immune to attack. Among the 112 monic irreducible polynomials in GF(7^3), only the first one is used in this paper. The different polynomial will produce different algorithm, hence a choice of polynomial can increase the security of the algorithm. Due to the randomness, the linear and differential cryptanalysis will be impossible for RDES cryptosystem. By observing the result, it is to be decided that other cryptographic algorithms may also be randomized by suitable incorporation of appropriate PRNG.

VII. ACKNOWLEDGMENT

We express our gratitude to the Principal of A.P.C. Roy Govt. College, University of North Bengal, Siliguri, Govt. of West Bengal for providing his moral support and infrastructural facilities towards research to the first author. We are also indeed thankful to the Head of the Department of Radio Physics and Electronics, University of Calcutta for providing necessary facilities to undertake research activities. Last but not least we are also grateful to the Head of the Department of Computer Science, A.P.C. Roy Govt. College for his necessary assistance in time management.

VIII. REFERENCES

[1] National Bureau of Standards, Data Encryption Standard, Federal Information Processing Standard Publication, Vol. 46, 1977.

[2] Stallings, W., “Cryptography and Network Security: Principles and practices (4th ed.)”, Pearson Education, 2008.

[3] Roelse, P., “The design of composite permutations with applications to DES-like S-boxes”, Design, Codes and Cryptography, Vol. 42, 2007, pp. 21-42.

[4] Rijmen V., “Cryptanalysis and Design of Block Ciphers”, A tutorial presentation in INDOCRYPT 2008.

[5] Lineham, A., and Gulliver, T. A., "Heuristic S-box Design", Contemporary Engineering Sciences, Vol. 1, No. 4, 2008, pp. 147-168.

[6] Wiener, M. J., "Efficient DES key search". Technical Report, TR-244, School of Computer Science, Carleton University, Ottawa, Canada, May 1994. Also presented in CRYPTO 1993.

[7] Zaman, JKMS., and Ghosh, R., "Search for Secure Random 8-bit Generator by Modular Approach of Statistical Test", Int. J. of Computer Applications, Vol. 96(10), pp. 32-41, 2014.

[8] Zaman, JKMS., and Ghosh, R., "A Pseudorandom Number Generator using Irreducible Polynomial over GF(7^3)", J. of Theoretical Physics & Cryptography, Vol. 12, pp. 18-25, 2016.

[9] Zaman, JKMS., and Ghosh, R., "Review on fifteen Statistical Tests proposed by NIST", J. of Theoretical Physics & Cryptography, Vol. 1, pp. 18-31, 2012.

[10] Rukhin A., Soto J., et al, A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications, NIST, US, Technology Administration, U.S. Department of Commerce, 2010.

[11] Rukhin A., Soto J., et al, A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications, NIST, US, Technology Administration, U.S. Department of Commerce 2008.

[12] Forouzan, B. A., "Cryptography and Network Security", Tata McGraw-Hill, New Delhi, Special Indian Ed., 2007.

[13] Kahate, A., "Cryptography and Network Security", 2nd. Ed., TMH New Delhi, 2008

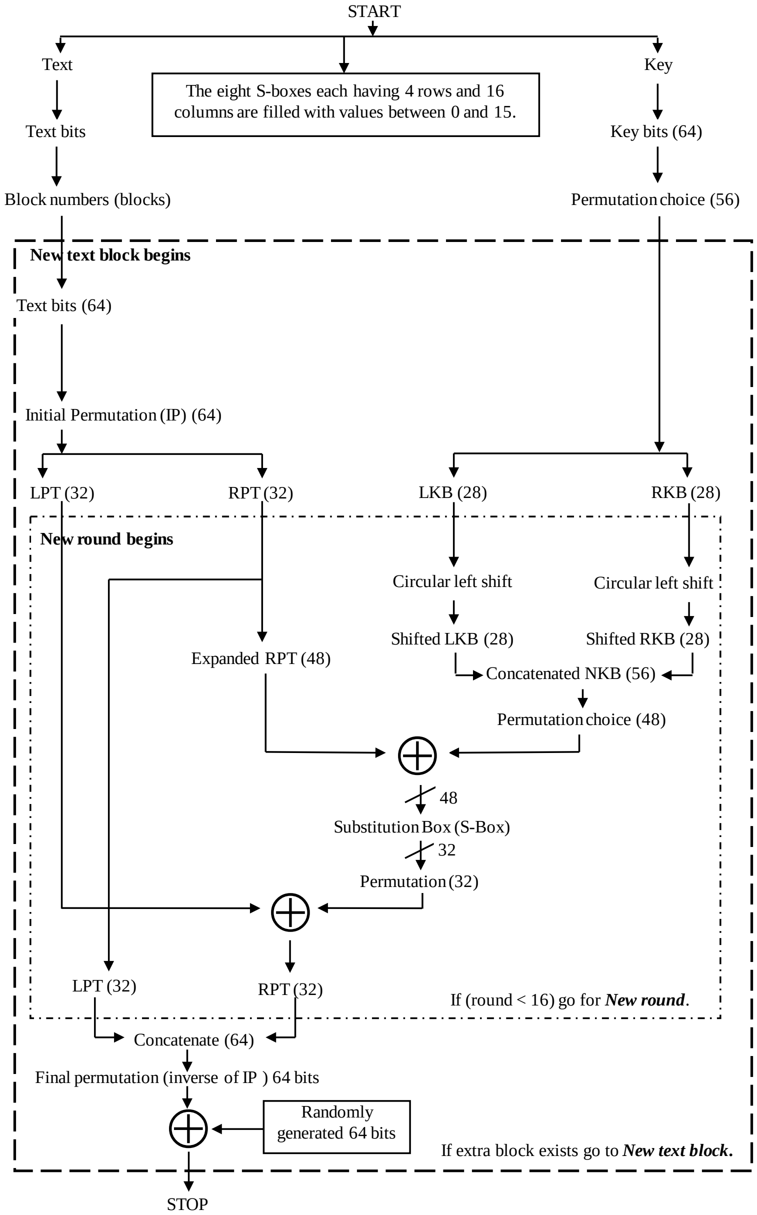
[14] Forouzan, B. A., "TCP/IP Protocol Suite", 3rd Ed., TMH, 2008.

[15] Mano, M. M., "Computer System Architecture", Pearson Education, 3rd Ed., 2004.

[16] Stinson, D. R., "Cryptography: Theory and Practice", 3rd Ed., Chapman & Hall, 2006.

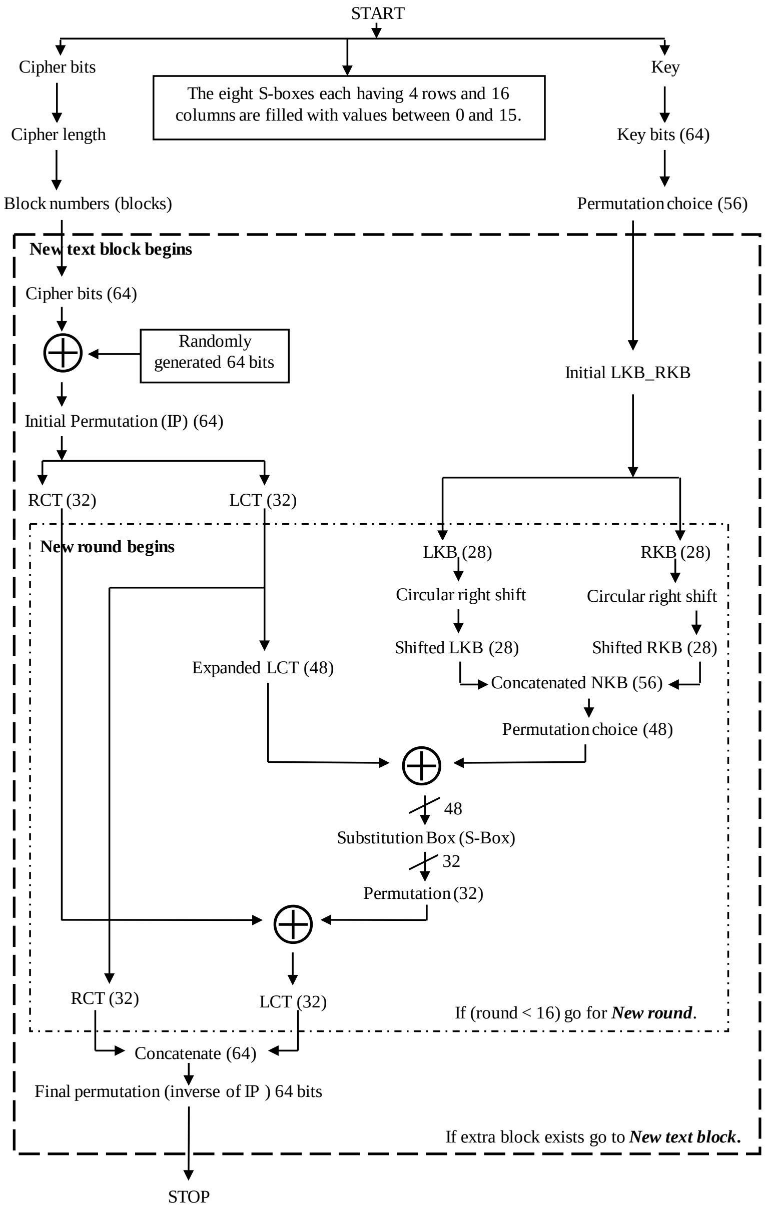
ANNEXURE-1

Data Flow Diagram of RDES Encryption Algorithm



ANNEXURE-2

Data Flow Diagram of RDES Decryption Algorithm


START
Cipher bits
The eight S-boxes each having 4 rows and 16 columns are filled with values between 0 and 15.
Key
Cipher length
Key bits (64)
Block numbers (blocks)
Permutation choice (56)
New text block begins
Cipher bits (64)
Randomly generated 64 bits
Initial LKB_RKB
Initial Permutation (IP) (64)
RCT (32)
LCT (32)
New round begins
LKB (28)
RKB (28)
Circular right shift
Circular right shift
Shifted LKB (28)
Shifted RKB (28)
Expanded LCT (48)
Concatenated NKB (56)
Permutation choice (48)
48
Substitution Box (S-Box)
32
Permutation (32)
RCT (32)
LCT (32)
If (round < 16) go for New round.
Concatenate (64)
Final permutation (inverse of IP) 64 bits
If extra block exists go to New text block.
STOP