DOI 10.12694/scpe.v20i1.1488
Scalable Computing: Practice and Experience ISSN 1895-1767
Volume 20, Number 1, pp. 161–180. http://www.scpe.org © 2019 SCPE

A SECURE STRUCTURE FOR HIDING INFORMATION IN A CRYPTOSYSTEM BASED ON MACHINE-LEARNING TECHNIQUES AND CONTENT-BASED OPTIMIZATION USING PORTFOLIO SELECTION DATA

CHANCHAL KUMAR[*]AND MOHAMMAD NAJMUD DOJA [†]

Abstract. Many systems including networks environment have higher complexity and ubiquitous connections than a normal system, hence design of a security system for hiding pertinent data a challenging task. This paper presents a secure structure that extends a protocol which is available for fast Diffe-Hellman protocol using Kummer surface. We show the extended version of the scheme by inclusion of an additional point, such that a more secure system can be constructed. The scheme has been build by employing machine-learning technique to select an appropriate class from multiple set of surfaces. A brief discussion on inclusion of multiple surfaces and making a selection of a specific surface using NSGA II algorithm is also provided. In this paper, we provide a brief overview of AES-128 (AES also known as Rijndael). In the starting, a short overview of the AES is given. This paper also has a description for altering the key generation module in AES based upon a newly designed content-based matrix which is built from portfolio selection data. The matrix is constructed using some predefined factors modifies the existing index which is computed based upon the context of the message. An optimization algorithm is employed for selecting specified entries from content matrix. These selected entries are used for altering the key generation algorithm in AES. The modified output obtained after altering the key generation scheme is provided in the paper. Lastly, a brief overview of LIM index, which is used as an index in cryptanalysis, is given. This paper has a description of the scheme to construct a more secure system that is capable of hiding the information with above-mentioned techniques.

Key words: Deffie-Hellman key exchange, Machine Learning techniques, NSGA II (Non-dominated Sorting Genetic Algorithm II), Key generation (AES) content based matrix, Optimization, LIM Index

AMS subject classifications. 68M12

1. Introduction.

The main objective of a cryptographic system may be stated as providing a capability for secure information exchange over a network. The plain text is the body of the message. Encryption is achieved by coding the input message using an algorithm along with a secret key. The coded message is called cipher-text. On the receiving side, the message is decoded using the secret key and reversing the steps of encryption algorithm. Thus, the plain text is obtained using the same secret key and decryption algorithm. The information which is encoded yields different sets of cipher. The study of information about structure of the cipher could be very beneficial for analysis. Cryptology deals with performing cryptanalysis in order to challenge the security of the encryption algorithm. Diffie-Hellman protocol is a commonly used protocol for secretly exchanging key [1]. In this protocol, the sender and receiver take a decision to select a finite set of points (Group) A, and choose a generator (v) from a subgroup. Using the private key (x_1) of the sender, a secret number $n_1 = (\alpha^{x_1} \bmod p)$ is generated. Here, p is a prime number and α is a primitive-root of p such that $\alpha < p$. The receiver generates a secret number $n_2 = (\alpha^{x_2} \bmod p)$ where x_2 is the private key of the receiver.The secret key exchange can be achieved by

$$k_1 = n_2^{x_1} \bmod p$$
$$k_2 = n_1^{x_2} \bmod p$$

Recently, a fast version of this protocol is provided that makes use of Kummer Surface [1, 23, 24]. In this paper, a new scheme is proposed for enhancing the security of the protocol by adding another point (u_5) on the surface. A brief overview of a scheme that can be incorporated for selecting one of the multiple surfaces using NSGA II is given in the paper.

1.1. A Brief Overview of AES.

During 1990's, it became clear that DES (Data Encryption Standard) will not be able to meet the required security standard [2]. In the year 1997, AES was adopted by NIST. AES was selected among other available ciphers. AES was chosen as a replacement for DES. AES now has a bigger key size and it is capable of handling the type of attacks meant for DES. In 2001, AES Rijndael was chosen

[*]Department of Computer Engineering, Jamia Millia Islamia, India (kumarchanchal943@gmail.com)
[†]Department of Computer Engineering, Jamia Millia Islamia, India (ndoja@yahoo.com)

among several other variants of ciphers. Previously DES was attacked using linear and differential cryptanalysis, these attacks were not successful in case of AES. AES uses block structure that means plain text is processed as blocks and the encryption is achieved on a block structure. AES composed of algebraic blocks which are relatively simple.

1.2. Different stages used in AES. AES is based upon block structure. The input plain text is divided in to various blocks of fixed size. The encryption takes place with each block separately. The size of the block is 128 bits. The allowed key size could be 128, 192 or 256 bits.

AES works with Galois Field (GF)(2^8), a polynomial $\left(b_7x^7 + b_6x^6 + ...b_0\right)$

is utilized for representing 8 bits (byte). Hexadecimal notation is used to represent bytes.

State: We are considering key size of 128 bits. The results are represented as state which is 128 bits in length and different operations are performed on the state. A matrix consisting of 4 rows and 4 columns is used as given below:

$$\begin{bmatrix} x(0,0) & x(0,1) & x(0,2) & x(0,3) \\ x(1,0) & x(1,1) & x(1,2) & x(1,3) \\ x(2,0) & x(2,1) & x(2,2) & x(2,3) \\ x(3,0) & x(3,1) & x(3,2) & x(3,3) \end{bmatrix}$$

The state is opterated with multiple rounds. Each round contains substitution box, Diffusion operation (Linear) and xoring with round key generated for each round.

The Diffusion operation consist of performing the shift rows operation then performing the transformation which is centred around mixing the different columns. For generating the round key for each round, the original key is used to generate a sub key for each round. Lastly, the algorithm used in Rijndael may described as below:

Firstly, the input message is placed in a state of 4×4 matrix. This state is xored with round key of 0^{th} round. Next, ten rounds are applied where the last round skips the mix column operation. The final state is the desired ciphertext.

This paper has description of a scheme that alters the key generation algorithm using a content-based matrix. The output of different keys for each round is given. The alteration in key generation is intended to enhance the hiding capability of AES and making it secure against different kinds of attacks.

1.3. Related Work. The use of theories of fuzzy extractor and secure sketch in the framework of key generation from biometrics is proposed in [4]. A description of lossy trapdoor function with their applications is given in [5, 6]. A significant use of pseudo-random number generators is presented in [7, 8]. A public key cryptosystem is more adaptive and secure against adaptive chosen ciphertext attack [9]. LIM (Lorenz Information Measure) based Cryptanalysis of AES-128 and AES-256 Block is presented in [11]. Some mathematical attacks like differential and linear cryptanalysis decrease the key search space but they not able to break the AES [12]. A description of some versions of Even-Mansour cipher scheme is given in [13]. Randomization based AES is better than first order differential electromagnetic and power analyses in term of low execution cost [14]. A combined approach of image processing and laser fault injections is given in [15] for security characterization of a hardware AES. A complete solution based on key updating framework to secure the execution of any kind of AES operation is proposed in [16]. A systematic toolbox is proposed in [17] for white-box implementation. A general study of the relationship between cryptography and machine learning is given in [18]. The primal machine algorithms are well performed in order to construct the encrypted models [19, 20, 21]. To perform successful key recovery deep learning techniques are appropriate [22]. A significant use of Kummer Surfaces is provided in [23, 24]. Symmetric and asymmetric cryptographic techniques based a new security protocol is proposed in [27] for online transaction.

2. Proposed System.

One of the vastly used algorithm for secure key exchange is Diffie-Hellman protocol. Kummer surface is recently proposed protocol for a fast-version of Diffie-Hellman protocol. This section provides description of the new protocol build using an additional point (u_5) in the extended surface. The details of the protocol are given below:

Algorithm 1 Modified Algorithm for Multiplication by N on the extended surface)

```
Input: U = $(U_1, U_2, U_3, U_4, U_5) \in$ extended surface and $N$
Output: $N \times U$ (multiplication on the extended surface)
Function 1: calculate $C_{jj}$ // (Output parameter $C_{jj}$)
Input: integers i,j;//(These are index parameters) float $U_v$, $U_w$ // ($U_v$ and $U_w$ are points on extended Kummer surface )
Output: value of $C_{jj}$;
Details: The output is computed using the following equation: $C_{jj}(U_v, U_w) = (U_j(V+W)* U_j(V-W) + (U_j(V-W) * U_j(V+W))$, where $U_i(V)$ denotes the ith component of $U_v = U(V)$
Function 2: calculate $C_{ij}$
Input: integers i,j; float $U_v$, $U_w$
Output:value of $C_{ij}$;
Details: The output is computed using the following equation: $(U_i(V+W)U_j(V-W) + U_i(V-W)U_j(V+W))$, for $1 \le i \le 4, 1 \le j \le 4$
Function 3: Add_on_extended_surface
Input: m = $(m_1, m_2, m_3, m_4$//(m is a point on extended Kummer surface))
integers $i, j$, float $U_v, U_w$
Output: Addition on extended surface
Details:The following steps are executed:
  1. $temp_1$ = calculate_$C_{ij}(i, j, *U_v, *U_w)$ where $temp_1$ and $temp_2$ are variables
  2. $temp_1$ = calculate_$C_{jj}(j, j, *U_v, *U_w)$
  3. Output = $(2 * m[j] * temp_1$ - $m[i] * temp_2$;)
  4. Parameters $q_p$ is initialized in the beginning of main routine and it is updated here. Parameter $t_9$ is a teporary variable which is initialized to 0, in the beginning of main routine.
     $x[i] = Output1, 1 \le i \le 4$
     if $(x[i] \neq 0))$ then calculate $q_p$
     $q_p = q_p - ((\frac{1}{x[i]}) + (\frac{1}{x[i]*x[i]})^2)$;
     $t_9 = t_9 + x[i] * q_p, 1 \le i \le 4$
     $t_9 = abs(t_9)$
     Parameter $ep_1$ is intialized in the beginning of main routine.
     Now, compute value of $x[5]$ using the following equation:
     $x[5] = t_9 + ep_1$;
     Parameters x,y and z are used in the main routine
     Here, x,y, and z = $(u_1, u_2, u_3, u_4, u_5)$
```

2.1. New Scheme adopted for inclusion of point (u_5) in the extended surface. The description of new method based on inclusion of point (u_5) in extended surface is presented here:

Need: A variant of Diffie-Hellman protocol based on Kummer surface is described in [5]. To enhance the security aspects of the structure, a new point (u_5) is being added. A sample run with of $(u_1, u_2, u_3, u_4, u_5)$ is provided in the paper.

Significance: The additional point (u_5) could prove a quite useful index for decision-making. The computed values of $(u_1, u_2, u_3, u_4, u_5)$ are used in a machine-learning algorithm to select a specific surface, in case, the design considers multiple surfaces and then chooses one surface for multiplication.

Impact: The selected surface based on computed value of (u_5) will be utilized to perform a multiplication. The output obtained with added point $(u_1, u_2, u_3, u_4, u_5)$ are given below. The details of main routine along with various functions used are given below:

2.1.1. Algorithm for Machine-learning for a sample data set of three extended surfaces. A sample technique for classification based on Machine-learning is described below given in algorithm 3.

Algorithm 2 Main routine

```
Input: N,x = (0,0,0,1,$u_5$)
y = $(u_1, u_2, u_3, u_4, u_5)$
z = $(u_1, u_2, u_3, u_4, u_5)$
Output: N*Point(x)
 1. Parameter: $m_1, i, j, ep_1, ep_2, q_p, t_9, t_{11}, t$;
 2. Initialize:
    $t_9 = 0; t_{11} = 0.0$; // (Temporary variables)
    $ep_1 = 0.00567; ep_2 = 0.00598; q_p = 0.06908$;//(Coefficients)
 3. Read the value of N if(N¡0) Then N = -N
    Initialize x($u_1 = 0, u_2 = 0, u_3 = 0, u_4 = 1, u_5 = 0.08$)
    Read the values of z $(u_1, u_2, u_3, u_4, u_5)$ and y $(u_1, u_2, u_3, u_4, u_5)$
 4. Initialize $t_9 = \sum_{i=1}^{4} \frac{x[i]}{4}; t_{11} = \sum_{i=1}^{4} \frac{y[i]}{4}$;
 5. Let h = H
 6. while $h \neq 0$ do
 7.     if $((h\%2)) \neq 0$ //If h is odd then
    $m_i = y[i]$; // $1 \leq i \leq 5$
 8.         for $(\ i = 1; i \leq 4; i++)$ do
    // Select value of j such that m[j] $\neq$ 0
    Calculate x[i] = add_on_extended_surface(m,i,j,x,z)
 9.         end forN = N - 1
10.     else
11.         $m_i = x[i]$; //$i \leq 1 \leq 5$
12.         for $(i = 1; i \leq 4; i++)$ do
    // Select value of j such that m[j] $\neq$ 0
    Calculate x[i] = add_on_extended_surface(m,i,j,x,z)
13.         end for
14.     end if
15.     $z[i] = 2 * z[i]$; //$1 \leq i \leq 4$
16.     $h = \frac{h}{2}$
17. end while
18. 7. Display the values of $x(u_1, u_2, u_3, u_4, u_5)$
```

2.2. Using NSGA II for selecting a particular surface. The Kummer surface is described by the equation [1]:

$$y = (f_6 \times x^6) + (f_5 \times x^5) + (f_4 \times x^4) + (f_3 \times x^3) + (f_2 \times x^2) + (f_1 \times x^1) + f_0 \tag{2.1}$$

The next section describes different sample surfaces. An application of NSGA II for formulating multi-objective functions using different combinations of these surfaces is given next. NSGA II is an example of evolutionary algorithm designed for multi-objective optimization [28]. NSGA II could also be employed to make the decision about a particular surface.The different multi-objective equations with corresponding outputs are described below.

2.2.1. Multi-Objective Program 1. The multi-objective program 1 contains objectives Z_1 and Z_2 for two selected surfaces, which uses a parameter ep_1. The output obtained by running NSGA II for these objectives is shown in Fig. 2.1.

The initial values of parameters and equations used for objectives are given below:

- Coefficients (f_i) used in surface 1 are given below:

$$f_0 = 100, f_1 = 5, f_2 = 0.8, f_3 = -1.87063, f_4 = 0.06, f_5 = 0.0, f_6 = 0.0, \text{ and } ep_1 = 6.089$$

Algorithm 3 Algorithm for Machine-learning for a sample data set of three extended surfaces

```
Step 1: Key = Data - $(u_1, u_2, u_3, u_4, u_5)$
Data contains 42 entries - surface 1, surface 2 and surface 3 contains 14 points
Key = Target ( surface 1 , surface 2 , surface 3)
// The algorithm given here has been tested for a sample data points of three extended surfaces.
Step 2: The in-built function train_test_split() , which is available in Python sklearn.model_selection
is used for training based on this data set.
The splitting is achived by the function, is given below:
X_train shape = 31, Y_train shape = 31,X_test shape = 11, Y_test shape = 11
The parameters X_train, Y_train, X_test and Y_test are used for splitting training and testing data points.
Step 3: KneighborsClassifier() function is used for classification
This function is available in sklearn.neighbors (Python)
Knn = Call KNeighborsClassifier()
Call Knn.fit() with X_train and Y_train parameters
State 4: Take a sample point x_new using numpy - arrays
A sample point → x_new(numpy - array) = [13.716864, 864.196899, 192.78521, 11. 769211]
This sample point has values of $(u_1, u_2, u_3, u_4, u_5)$
Step 5: Call predict() function with x_new point as input parameter
The output obtained is index=1, which is, surface number = 2.
Here, Index 0: Surface 1, Index 1: Surface 2, Index 2: Surface 3
This algorithm may be utilized for selecting a specific surface from a set of multiple surfaces.
```

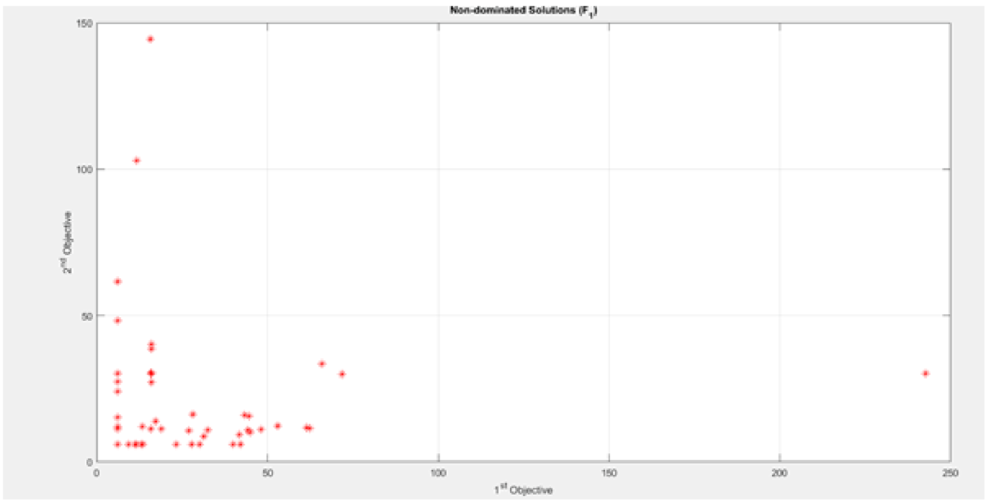


FIG. 2.1. *Output of Multi-Objective Program 1*

- Coefficients (f_i) used in surface 2 are given below:

$$f_{02} = 120, f_{12} = 3, f_{22} = 0.65, f_{32} = -1.87063, f_{42} = 0.08, f_{52} = 0.09, f_{62} = 0.00085$$

- Objective 1:

$$z_1 = ep_1 + \sqrt{((f_6 \times x^6) + (f_5 \times x^5) + (f_4 \times x^4) + (f_3 \times x^3) + (f_2 \times x^2) + (f_1 \times x^1) + f_0)} \tag{2.2}$$

- Objective 2:

$$z_2 = \sqrt{((f_{62} \times x^6) + (f_{52} \times x^5) + (f_{42} \times x^4) + (f_{32} \times x^3) + (f_{22} \times x^2) + (f_{12} \times x^1) + f_{02})} \tag{2.3}$$

2.2.2. Multi-Objective Program 2. The multi-objective program 2 contains objectives Z_1 and Z_2 for two selected surfaces, which uses a parameter ep_1.The parameter of f_1 of surface 1 has been changed here, as compared to multi-objective program 1. The output obtained by running NSGA II for these objectives is shown in Fig.2.2.

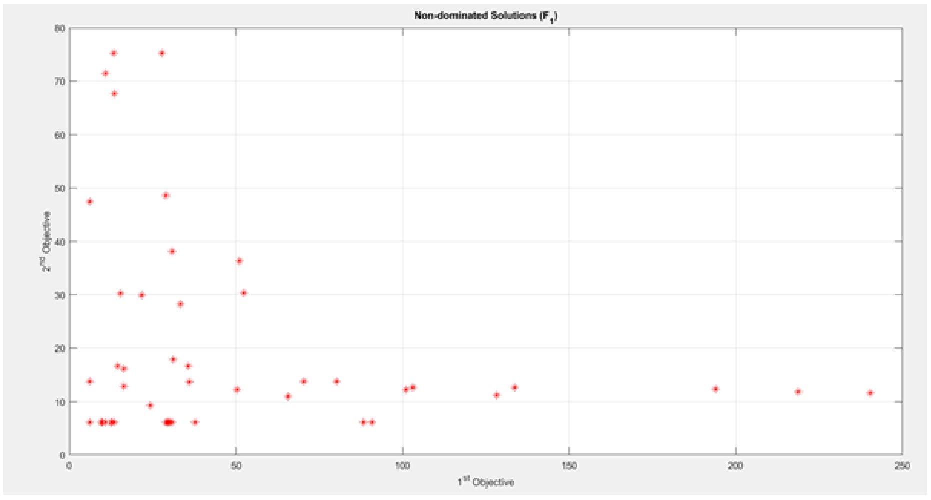


FIG. 2.2. *Output of Multi-Objective Program 2*

The initial values of parameters and equations used for objectives are given below:
- Coefficients (f_i) used in surface 1 are given below:

$$f_0 = 100, f_1 = 0.0, f_2 = 0.8, f_3 = -1.87063, f_4 = 0.06, f_5 = 0.0, f_6 = 0.0, \text{ and } ep_1 = 6.089$$

- Coefficients (f_i) used in surface 2 are given below:

$$f_{02} = 120, f_{12} = 3, f_{22} = 0.65, f_{32} = -1.87063, f_{42} = 0.08, f_{52} = 0.09, f_{62} = 0.00085$$

- Objective 1:

$$z_1 = ep_1 + \sqrt{((f_6 \times x^6) + (f_5 \times x^5) + (f_4 \times x^4) + (f_3 \times x^3) + (f_2 \times x^2) + (f_1 \times x^1) + f_0)} \tag{2.4}$$

- Objective 2:

$$z_2 = \sqrt{((f_{62} \times x^6) + (f_{52} \times x^5) + (f_{42} \times x^4) + (f_{32} \times x^3) + (f_{22} \times x^2) + (f_{12} \times x^1) + f_{02})} \tag{2.5}$$

2.2.3. Multi-Objective Program 3. The multi-objective program 3 contains objectives Z_1 and Z_2 for selected surface, which uses a parameter ep_1 . Here, objective 1 uses two different parameters q_1 and M_i , it is described by equation (6). The output obtained by running NSGA II for these objectives is shown in Fig. 2.3.

The initial values of parameters and equations used for objectives are given below:
- $ep_1 = 6.089, M_i = 120, q_1 = 7.41$
- Coefficients (f_i) used in surface 2 are given below:

$$f_{02} = 120, f_{12} = 3, f_{22} = 0.65, f_{32} = -1.87063, f_{42} = 0.08, f_{52} = 0.09, f_{62} = 0.00085$$

- Objective 1:

$$z_1 = abs(ep_1 + q_1 \times M_i^2 + q_1 \times M_i \times x^2) \tag{2.6}$$

- Objective 2:

$$z_2 = \sqrt{((f_{62} \times x^6) + (f_{52} \times x^5) + (f_{42} \times x^4) + (f_{32} \times x^3) + (f_{22} \times x^2) + (f_{12} \times x^1) + f_{02})} \tag{2.7}$$

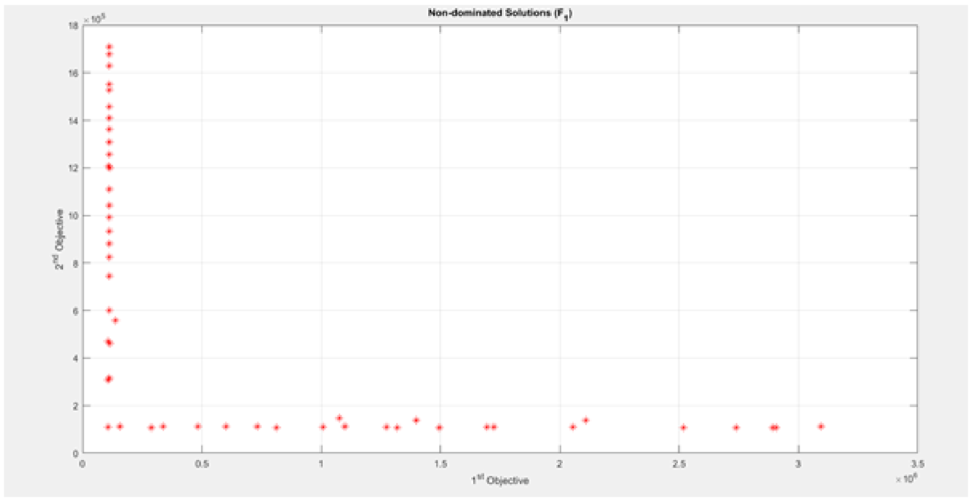


FIG. 2.3. *Output of Multi-Objective Program 3*

2.2.4. Multi-Objective Program 4. The multi-objective program 4 contains objectives Z_1 and Z_2 for selected surface, which uses a parameter ep_1. Here, objective 1 uses two different parameters q_1 and M_i, it is described by equation (8) and objective 2 has different parameters, as compared to multi-objective program 3. The output obtained by running NSGA II for these objectives is shown in Fig. 2.4.

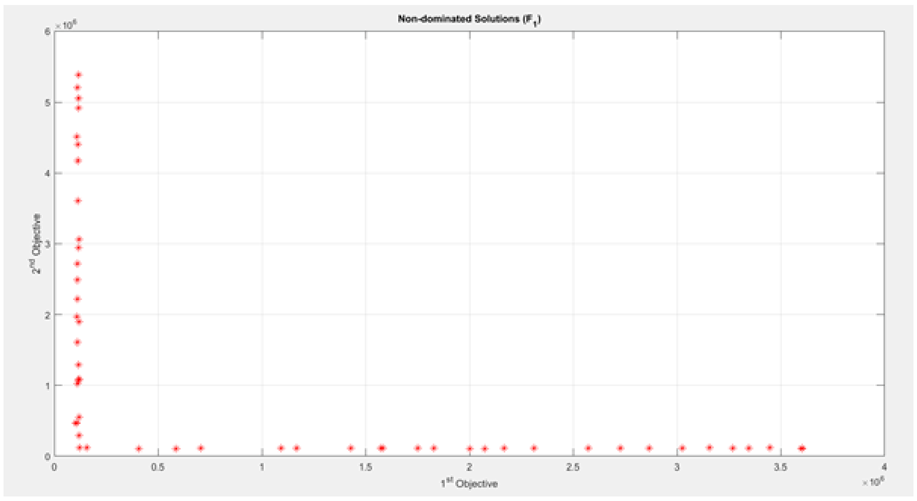


FIG. 2.4. *Output of Multi-Objective Program 4*

The initial values of parameters and equations used for objectives are given below:

- Coefficients (f_i) used in surface 1 are $f_0 = 100, f_1 = 0.0, f_2 = 0.8, f_3 = -1.87063, f_4 = 0.06, f_5 = 0.0, f_6 = 0.0$ and $ep_1 = 6.089, M_i = 120, q_1 = 7.41$
- Objective 1:

$$z_1 = abs(ep_1 + q_1 \times M_i^2 + q_1 \times M_i \times x^2) \tag{2.8}$$

- Objective 2:

$$z_2 = \sqrt{((f_{62} \times x^6) + (f_{52} \times x^5) + (f_{42} \times x^4) + (f_{32} \times x^3) + (f_{22} \times x^2) + (f_{12} \times x^1) + f_{02})} \tag{2.9}$$

TABLE 2.1
Sample input string composed of data streams of portfolio selection data that is used in SHA1 algorithm

S.No.	Input String used in SHA1 algorithm				
	Expected Return	Risk	Parameter T_a	Parameter B_5	Parameter B_{10}
1.	“0.2560	0.1622	0.023	0.5628	0.4372” +
	“0.2786	0.1641	0.023	0.5002	0.4998” +
	“0.3012	0.1698	0.023	0.4377	0.5623” +
	“0.3238	0.1788	0.023	0.3752	0.6248” +
	“0.3464	0.1907	0.023	0.3126	0.6874” +
	“0.3690	0.2050	0.023	0.2501	0.7499” +
	“0.3916	0.2050	0.023	0.1876	0.8124” +
	“0.4142	0.2390	0.023	0.1251	0.8749” +
	“0.4368	0.2579	0.023	0.0438	0.9259” +
	“0.4594	0.2779	0.023	0.0	1.0”

2.3. Using SHA-1 for authentication . For authentication SHA-1 algorithm could be utilized and a sample run with portfolio data is given in Table 2.1. Where, parameter T_a and parameters B_5, B_{10} are different values which are used in portfolio selection problem.
The output obtained after giving the input string, which is described in Table 2.1, to SHA1 algorithm is “DCF5C49BB3160A70788A72A06318FB2BE69BB233”.

2.4. Cryptanalysis of the proposed protocol. It is proved that discrete logarithm solution for Kummer surface is equivalent to discrete logarithm solution for Jacobian [1]. With the additional point u_5 the security of the proposed protocol has not been lost moreover an additional direction for ensuring security has been included. The intension is to make the proposed protocol more secure with the help of new point u_5. The encryption scheme based on proposed protocol may utilize machine-learning methods for selecting a specified surface in the case of multiple surfaces. The machine learning algorithm has been tested on a sample data of three extended surfaces. As suggested in [1], ELGamal scheme for encryption may be designed that is based on calculation on the Kummer surface. The addition function provided here could be used for multiplication on the extended surface. Thus, the use of extended surface does not account for any security losses.

3. Generating keys of each round of AES using content-based matrix. This section provides details about the new scheme that is being used for key generation (AES). An overview of the structure of content-based matrix using sample data of portfolio selection is also given in Fig 3.1.

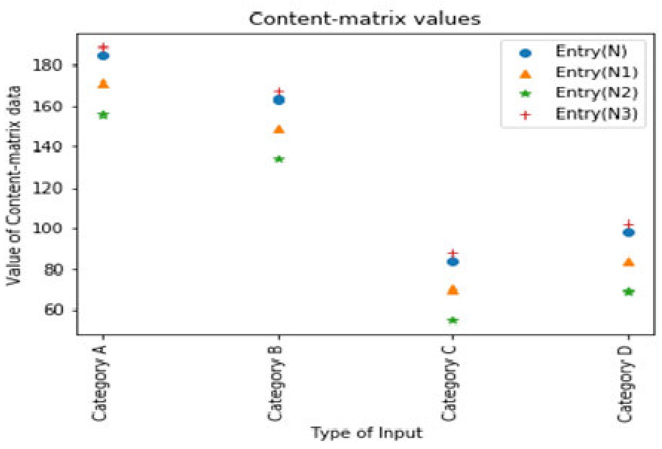


FIG. 3.1. *Content-matrix values*

The formulation of the content-matrix is given next. Here, the input is divided into four categories- category A for characters a-z / A-Z, category B for numbers 0-9, category C for special characters and lastly, the category D for images / videos etc.

Here, a sample frequency distribution of different classes in a sample input is given in Table 3.1. It contains assumed values of weight (x) and the mapping output (y) is found using the following equation:

$$y = x^3 + x^2 + w$$
$$N_o = y \bmod 255$$

The following constants are used here: epk_1 = 13.89, epk_2 = 29.17 and epk_3 = - 4.089. The values of N_1, N_2 and N_3 are computed as described below:

$$N_1 = N - epk_1$$
$$N_2 = N - epk_2$$
$$N_3 = N - epk_3$$

Table 3.1
Output of content-matrix

S.No.	Type of input	Frequency(x)	Weight(w)	Mapping output(y)	Entry (N_0)	Entry(N_1)	Entry(N_2)	Entry(N_3)
1	Category A	34	15	40475	185	171.11	155.825	189.089
2	Category B	50	8	6252508	163	149.11	133.825	167.089
3	Category C	60	9	216069	84	70.11	54.825	88.089
4	Category D	12	11	1883	98	84.11	68.825	102.089

The reduced content-matrix values are based upon only three categories: category-$alpha_1$ =category 1, category-$alpha_2$ = category 2 and category-$alpha_3$ = category 3. The output obtained are given in Table 3.2 and Fig. 3.2.

Table 3.2
Output of content-matrix

S.No.	Type of input	Entry(N)	Entry(N_1)	Entry(N_2)	Entry(N_3)
1	Category-α_1	185	171.11	155.825	189.089
2	Category-α_2	163	149.11	133.825	167.089
3	Category-α_3	91	77.11	61.825	95.089

The following equations are used to compute optimal values :

$$cost_1 = a_1\alpha_1^2 + b_1\alpha_1 + c_1 \tag{3.1}$$
$$cost_2 = a_2\alpha_2^2 + b_2\alpha_2 + c_2 \tag{3.2}$$
$$cost_3 = a_3\alpha_3^2 + b_3\alpha_3 + c_3 \tag{3.3}$$
$$\texttt{Total\_cost} = cost_1 + cost_2 + cost_3 \tag{3.4}$$
$$189.089 \leq a_1 \leq 155.825$$
$$167.089 \leq a_2 \leq 133.825$$
$$95.089 \leq a_3 \leq 61.825$$
$$461.0 \leq \texttt{Total\_cost} \leq 361.0$$

The optimized values obtained after running classical Lagrangian multiplier method with sample input-values of a_i, b_i and c_i and using the inequality constraints values of a_1, a_2 and a_3 described above, are presented in Table 3.3 and Fig. 3.3.

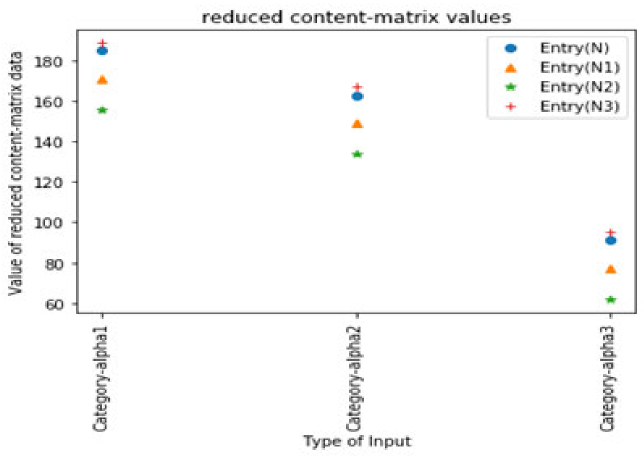


FIG. 3.2. *Reduced content-matrix values*

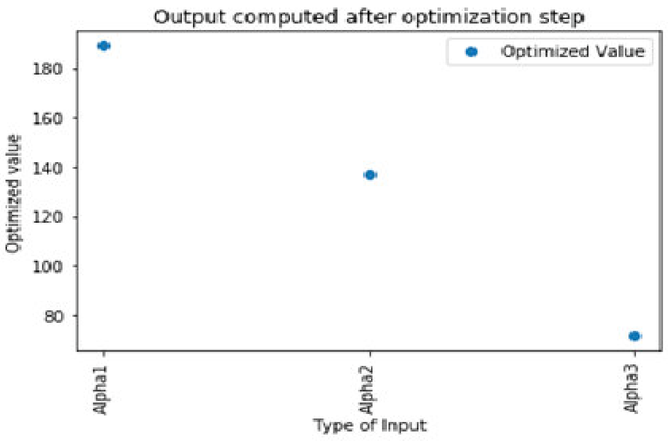


FIG. 3.3. *Output computed after optimization step*

TABLE 3.3
Optimized values of different categories

S.No.	Category	Optimized value
1	Category-α_1	189.0809
2	Category-α_2	137.2910
3	Category-α_3	71.6765

3.1. Cryptanalysis of the proposed scheme. Different kinds of cryptanalysis for AES (Rijndael) is provided in [28]. Using another technique based on calculating partial sum, the complexity of AES can be decreased [28]. The attacks on 7 and 8 rounds can be achieved by incorporating the information about extra texts. The next area to be considered is Key generation. Various undesired behaviour for key generation is discussed in [28].

This paper has made presentation about a novel scheme that may be adopted to conceal the information which could prove fruitful in increasing complexity of Key generation. Specifically, the structure of content-based matrix and selecting optimized values play a vital role in hiding the information. Since Key generation has a small number of non-linear components, the schemes presented here has given new dimension in this direction.

Algorithm 4 New Key Expansion Algorithm

```
1: The new modified key expansion algorithm is given below:
2: Modified-Key-Expansion(byte x[4*y], word out[4 * 11], y)
3: start
4: word $t_{11}$ // ( temporary variable)
5: j= 0
6: While (j ¡ y)
7: out [i] = word(x[4 * j], key[4 * j+1], x[4 * j+2], x[4 * j+3]
8: j = j+1
9: end of while
10: j = y
11: While (j ¡ (4 * 11))
12: $t_{11}$ = out [j-1] // stage 1
13: if (j mod y == 0) then
14:     $t_{11}$ = Rotate_Word(temp) // stage 2
15:     $t_{11}$ = Substitute_Word(temp) // stage 3
16:     $t_{11}$ = ($t_{11}$ xor content matrix_data_entry) // stage 4
17:     $t_{11}$ = $t_{11}$ xor R_constant[j/4] // stage 5
18: elseif (y ¿ 6 and j mod y = 4)
19:     $t_{11}$ = Substitute_Word(temp) // stage 6
20: end if
21: out[j] = out[j-y] xor $t_{11}$ // stage 7 and stage 8
22: j = j + 1
23: end of while
24: end
```

4. Result Analysis and Discussion. The following section describes the various outputs obtained after executing extended Kummer surface algorithm explained in section 2.1. An overview of the results obtained for existing Kummer surface is also given. The next section has a description of data obtained after executing new key expansion (AES) algorithm. Lastly, an overview of the results obtained with LIM algorithm are provided.

4.1. Existing Kummer Surface Results. This section briefly presents results which are obtained for existing and extended Kummer surface. The results are given in Table 4.1 and Fig 4.1 for existing Kummer surface- surface 1.

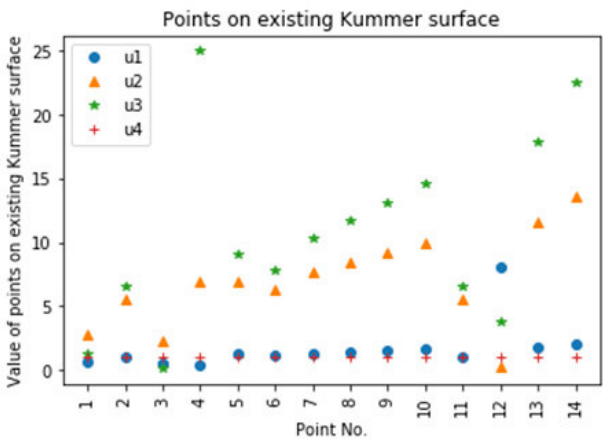


FIG. 4.1. *Points on existing Kummer surface (Surface 1)*

TABLE 4.1
Sample points on Existing Kummer Surface(u_1, u_2, u_3, u_4, u_5)(Surface 1)

S.No.	u_1	u_2	u_3	u_4
1	0.5915	2.7999	1.2000	1.0000
2	1.0000	5.5856	6.5497	1.0000
3	0.5229	2.2824	0.1000	1.0000
4	0.4217	6.8683	25.0000	1.0000
5	1.2000	6.9648	9.0642	1.0000
6	1.1000	6.2700	7.7984	1.0000
7	1.3000	7.6731	10.3607	1.0000
8	1.4000	8.3982	11.7018	1.0000
9	1.5000	9.1445	13.1036	1.0000
10	1.6000	9.9176	14.5872	1.0000
11	1.0000	5.5856	6.5460	1.0000
12	8.0000	0.2864	3.7636	1.0000
13	1.8000	11.5840	17.9428	1.0000
14	2.0000	13.6073	22.5198	1.0000

Similarly, sample data points are computed for surface 2 and surface 3.

4.1.1. Extended Kummer Surface Results. This section presents the results which are obtained after executing extended Kummer surface algorithm. This algorithm contains a newly added point (u_5). The input points z and y on the extended Kummer surface (surface 1) are given in Table 4.2 and are drawn in Fig. 4.2 and Fig. 4.3.

TABLE 4.2
Value of input point (z) and input point (y) on extended Kummer surface (Surface 1)

S.No.	z(u_1)	z(u_2)	z(u_3)	z(u_4)	z(u_5)	y(u_1)	y(u_2)	y(u_3)	y(u_4)	y(u_5)
1	1.0000	5.5856	6.5460	1.0000	0.8000	8.0000	0.2864	3.7636	1.0000	0.9800
2	1.1000	6.2700	7.7984	1.0000	0.8000	1.3000	7.6731	10.3607	1.0000	0.9800
3	1.2000	6.9648	9.0642	1.0000	0.8000	1.1000	6.2700	7.7984	1.0000	0.9800
4	0.5915	2.7999	1.2000	1.0000	0.8000	1.0000	5.5856	6.5497	1.0000	0.9800
5	8.0000	0.2864	3.7636	1.0000	0.8000	2.5033	13.0410	5.0000	1.0000	0.9800
6	1.5000	9.1445	13.1036	1.0000	0.8000	1.6000	9.9176	14.5872	1.0000	0.9800
7	0.4217	6.8683	25.0000	1.0000	0.8000	1.2000	6.9648	9.0642	1.0000	0.9800
8	1.4000	8.3982	11.7018	1.0000	0.8000	1.5000	9.1445	13.1036	1.0000	0.9800
9	0.5229	2.2824	0.1000	1.0000	0.8000	0.4217	6.8683	25.0000	1.0000	0.9800
10	1.6000	9.9176	14.5872	1.0000	0.8000	1.0000	5.5856	6.5497	1.0000	0.9800
11	1.0000	5.5856	6.5497	1.0000	0.8000	0.5229	2.2824	0.1000	1.0000	0.9800
12	1.8000	11.5840	17.9428	1.0000	0.8000	2.0000	13.6073	22.5198	1.0000	0.9800
13	1.3000	7.6731	10.3607	1.0000	0.8000	1.4000	8.3982	11.7018	1.0000	0.9800
14	2.0000	13.6073	22.5198	1.0000	0.8000	0.5915	2.7999	1.2000	1.0000	0.9800

Output points obtained are given in Table 4.3 and drawn in Fig.4.4. The data points obtained are sorted output according to point (u_5). Similar outputs obtained for surface 2 and surface 3. Moreover, the data points are sorted according to point (u_5). An effort is made to present output points pictorially in ascending order of point (u_5) such that the variation of point (u_5) can be studied with values of other points (u_1, u_2, u_3, u_4). A meaningful inference about the variation of point (u_5) can be achieved with large set of input points.

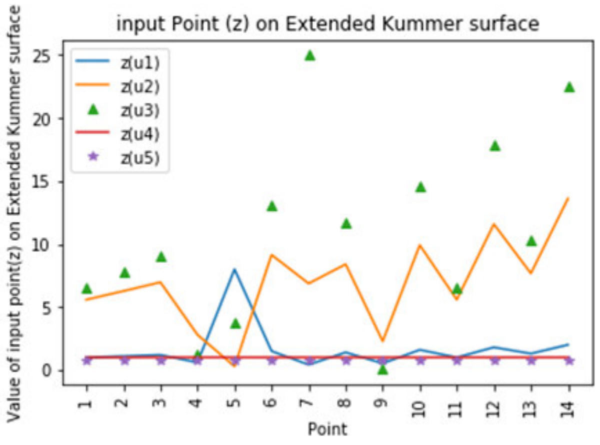


Fig. 4.2. *Input point (z) on Extended Kummer surface (surface 1)*

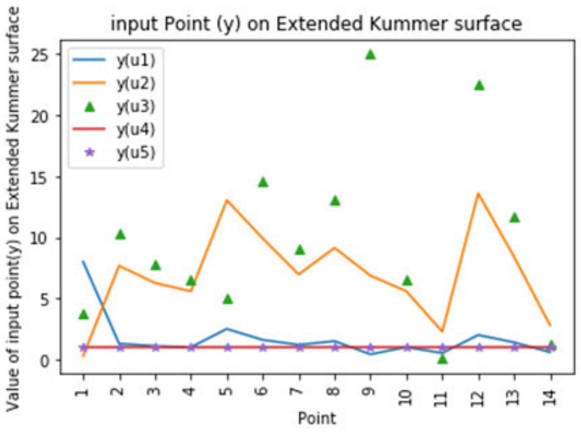


Fig. 4.3. *Input point (y) on Extended Kummer surface (surface 1)*

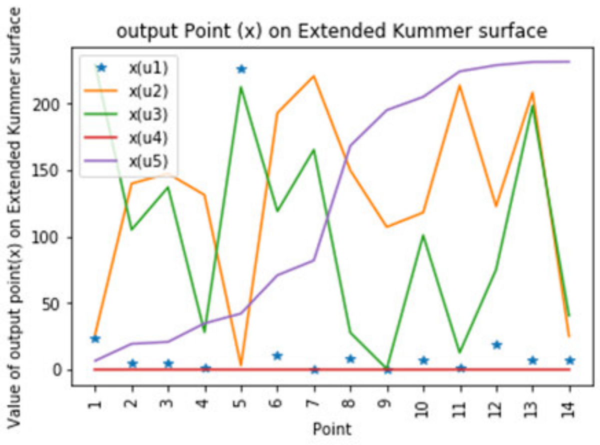


Fig. 4.4. *Output point (x) Extended Kummer surface (surface 1)*

Table 4.3
Output values of extended Kummer Surface($x(u_1), x(u_2), x(u_3), x(u_4), u_5$)(Surface 1)

S.No.	$x(u_1)$	$x(u_2)$	$x(u_3)$	$x(u_4)$	$x(u_5)$
1	24.0000	26.8055	229.3580	0.0000	6.7970
2	4.7190	139.9495	105.2500	0.0000	19.4924
3	4.7520	147.4385	137.1388	0.0000	20.9641
4	1.0496	131.3605	28.2947	0.0000	34.8777
5	225.6336	3.2089	212.4693	0.0000	42.2683
6	10.8000	192.9697	119.0908	0.0000	71.0500
7	0.6403	220.6695	165.3555	0.0000	82.1640
8	8.8200	149.8854	27.9238	0.0000	168.2817
9	0.3459	107.3347	0.7500	0.0000	195.0867
10	7.6800	118.1619	101.0610	0.0000	205.0549
11	1.5687	213.6205	12.8696	0.0000	224.2009
12	19.4400	122.8515	75.2773	0.0000	228.8854
13	7.0979	208.3531	198.3577	0.0000	231.2859
14	7.0981	25.2546	40.7057	0.0000	231.4460

4.2. Results obtained with new Key expansion algorithm in different stages. This section provides an overview of the outputs obtained after different stages in new Key expansion algorithm. For simplicity of the presented output, the decimal values on a scale of (value/10^9) is chosen. Table 4.4 has output of stage 4 and its (new Key expansion) diagram along with two trend lines - trade line 1(power trend line with forecast of 2.0 periods, equation used: y = $34092 \times x^{00743}$, R^2 = 0.1335, here y is output, x is input parameter for trend line and R^2 is R-squared value used for trend line 1) and trend line 2 (polynomial trend with forecast of 2.0 periods) is given in Fig.4.5.

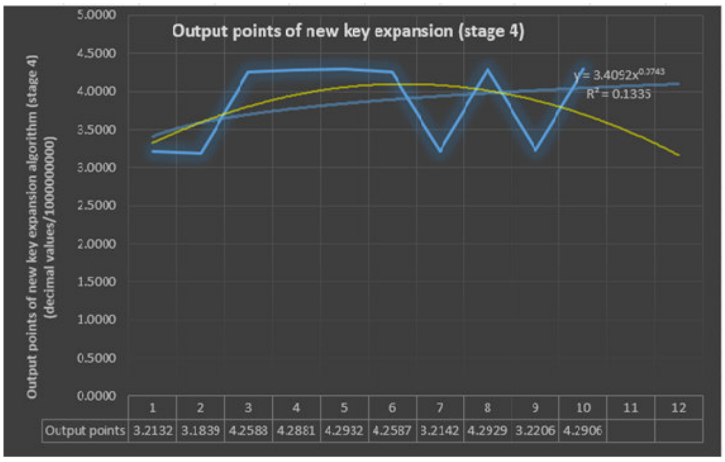
Table 4.4
Output data obtained from new Key expansion algorithm: Stage 4 and Stage 6

S. No.	O(hex):Stage 4	O(dec):Stage 4	O(hex):Stgae 6	O(dec):Stage 6
1	bf85ef55	3.2132	be85ef55	3.1964
2	bdc5d7d4	3.1839	bfc5d7d4	3.2174
3	fdd76f5f	4.2588	f9d76f5f	4.1916
4	ff97ef7d	4.2881	f797ef7d	4.1539
5	ffe5c7ff	4.2932	efe5c7ff	4.0248
6	fdd5e6dc	4.2587	ddd5e6dc	3.7218
7	bf954f5e	3.2142	ff954f5e	4.2880
8	ffdfd75d	4.2929	7fdfd75d	2.1454
9	bff5ee5c	3.2206	a4f5ee5c	2.7676
10	ffbdcf74	4.2906	c9bdcf74	3.3847

Table 4.4 has also output of stage 6 and its (new Key expansion) diagram along with two trend lines - trade line 1(power trend line with forecast of 2.0 periods, equation used: y = $3.7022 \times x^{0.049}$, R^2 = 0.0266) and trend line 2 (polynomial trend with forecast of 2.0 periods) is given in Fig.4.6.

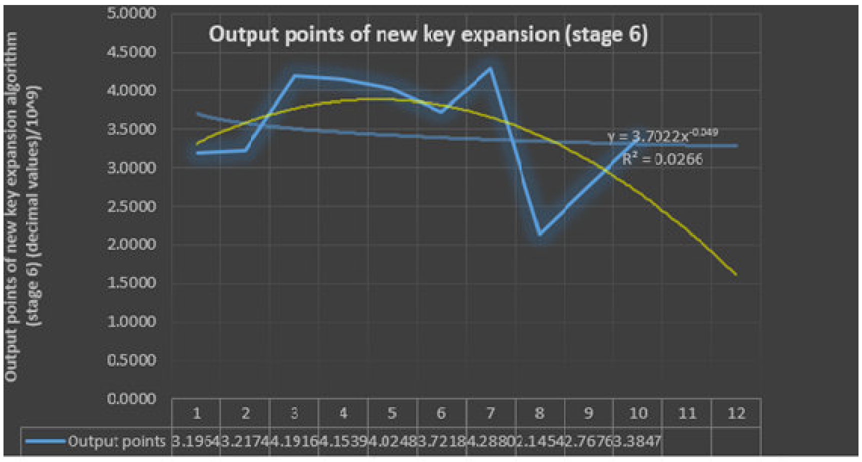
In Table 4.4, O(hex):Stage 4 denotes Output of new Key expansion algorithm: stage 4 (Hex value), O(dec):Stage 4 denotes Output of new Key expansion algorithm : stage 4 (Decimal value / 10^9), O(hex):Stage 6 denotes Output of new Key expansion algorithm : stage 6 (Hex value), and O(dec):Stage 6 denotes Output of new Key expansion algorithm : stage 6 (Decimal value / 10^9).

In Table 4.5 ,O(hex):Stage 7 denotes Output of new Key expansion algorithm : stage 7 (Hex value), O(dec):Stage 7 denotes Output of new Key expansion algorithm : stage 7 (Decimal value / 10^9), O(hex):stage



	1	2	3	4	5	6	7	8	9	10	11	12
Output points	3.2132	3.1839	4.2583	4.2881	4.2932	4.2587	3.2142	4.2929	3.2206	4.2906		

FIG. 4.5. *Output points of new key expansion (stage 4)*



	1	2	3	4	5	6	7	8	9	10	11	12
Output points	3.1954	3.2174	4.1916	4.1539	4.0248	3.7218	4.2880	2.1454	2.7676	3.3847		

FIG. 4.6. *Output points of new key expansion (stage 6)*

8 denotes Output of new Key expansion algorithm : stage 8 (Hex value), and O(dec):stage 8 denotes Output of new Key expansion algorithm : stage 8 (Decimal value / 10^9).

Table 4.5 also has output of stage 7 and its (new Key expansion) diagram along with two trend lines trade line 1(power trend line with forecast of 2.0 periods, equation used: y = $0.9656 \times x^{0.0177}$, $R^2 = 0.0002$) and trend line 2 (polynomial trend with forecast of 2.0 periods) is given in Fig.4.7.

Table 4.5 has output of stage 8 and its (new Key expansion) diagram along with two trend lines-trade line 1(power trend line with forecast of 2.0 periods, equation used: y = $0.9656 \times x^{0.0177}$, $R^2 = 0.0002$) and trend line 2 (polynomial trend with forecast of 2.0 periods) is given in Fig.4.8.

As evident from the outputs presented in Table 4.4 and Table 4.5, the trade lines are having different patterns in the existing and new Key expansion algorithm. The output of stage 1, 2, 3 and 5 have same outputs in both existing and new Key expansion algorithm, and are given in Table 4.6. The stage 4 is available only in the new Key expansion. The outputs in the stage 6 are altogether different in new algorithm. The variation of points is different in new stage 7, whereas the trend lines are having similar patterns in this stage. Lastly, the stage 8 has different variation of points in new Key expansion, moreover trade lines depicts different behaviour. Thus, the proposed scheme adopted can be utilized by a designer in order to provide better security by having newer variation patterns, which are different from existing AES Key expansion algorithm.

Table 4.5
Output data obtained from new Key expansion algorithm: Stage 7 and Stage 8

S.No.	O(hex):Stage 7	O(dec):Stage 7	O(hex):Stage 8	O(dec):Stage 8
1	2b7e1576	0.7297	2b7e1576	0.7297
2	28aed2a6	0.6825	28aed2a6	0.6825
3	abf71588	2.8851	abf71588	2.8851
4	09cf4f3c	0.1646	09cf4f3c	0.1646
5	95fbfa43	2.5163	95fbfa43	2.5163
6	bd5528e5	3.1765	bd5528e5	3.1765
7	16a23d6d	0.3797	16a23d6d	0.3797
8	1f6d7251	0.5273	1f6d7251	0.5273
9	2a3e2d97	0.7087	2a3e2d97	0.7087
10	976b572	0.1588	976b572	0.1588
11	81c9381f	2.1774	81c9381f	2.1774
12	9ea44a4e	2.6616	9ea44a4e	2.6616
13	d3e942c8	3.5553	d3e942c8	3.5553
14	448247ba	1.1494	448247ba	1.1494
15	c54b7fa5	3.3101	c54b7fa5	3.3101
16	5bef35eb	1.5424	5bef35eb	1.5424
17	247eadb5	0.6123	247eadb5	0.6123
18	60fceaf	0.1017	60fceaf	0.1017
19	a5b795aa	2.7803	a5b795aa	2.7803
20	fe58a041	4.2672	fe58a041	4.2672
21	cb9b6a4a	3.4160	cb9b6a4a	3.4160
22	ab678045	2.8757	ab678045	2.8757
23	ed015ef	0.2485	ed015ef	0.2485
24	f088b5ae	4.0355	f088b5ae	4.0355
25	164e8c96	0.3742	164e8c96	0.3742
26	bd29cd3	0.1984	bd29cd3	0.1984
27	b3f9193c	3.0194	b3f9193c	3.0194
28	4371ac92	1.1315	4371ac92	1.1315
29	e9dbc3c8	3.9235	e9dbc3c8	3.9235
30	54f2cf1b	1.4252	54f2cf1b	1.4252
31	e7bd627	0.2430	e7bd627	0.2430
32	a47a7ab5	2.7595	a47a7ab5	2.7595
33	9641495	0.1576	9641495	0.1576
34	c2f6db8e	3.2710	c2f6db8e	3.2710
35	25fdda9	0.0398	25fdda9	0.0398
36	8187771c	2.1731	8187771c	2.1731
37	32f1fac9	0.8547	32f1fac9	0.8547
38	f072147	0.2521	f072147	0.2521
39	d5fa2cee	3.5899	d5fa2cee	3.5899
40	547d5bf2	1.4175	547d5bf2	1.4175

4.3. Output of LIM. This section presents computed output values of LIM obtained from histogram data taken from different extended Kummer surface. The values of LIM for data used in extended Kummer surface 1 is given in Table 4.7, the diagrams showing histograms for first five values (set 1) and last five values (set 2) of Table 4.7 are given in Fig.4.9 and Fig.4.10. The computed values of LIM for surface 1 are depicted in Fig. 4.11. The algorithm of LIM is presented in [10].

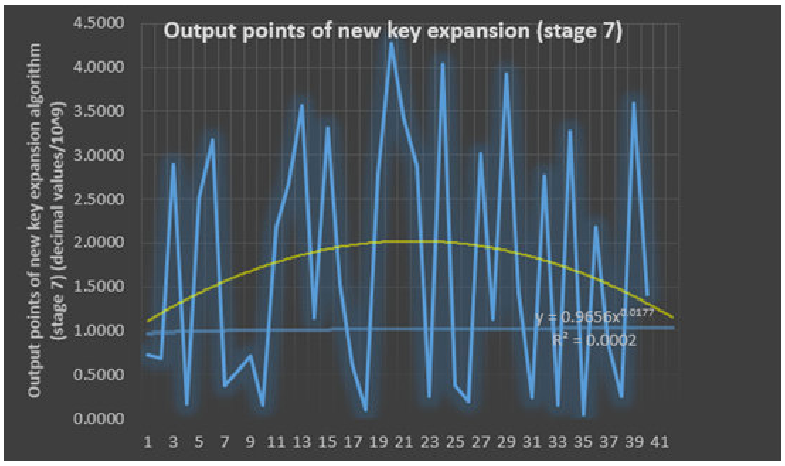


FIG. 4.7. *Output points of new key expansion (stage 7)*

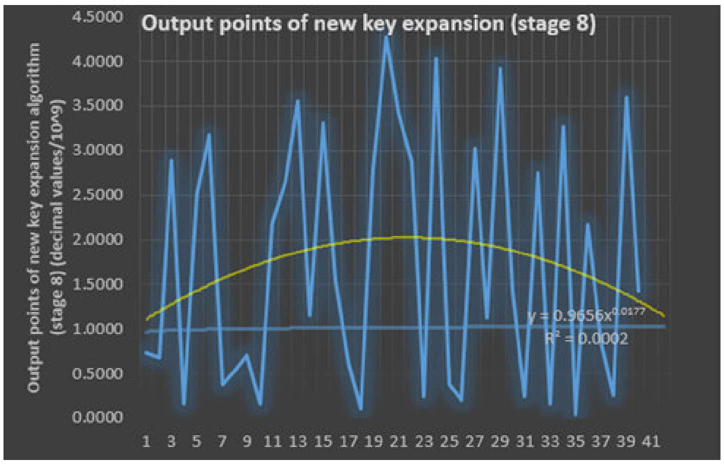
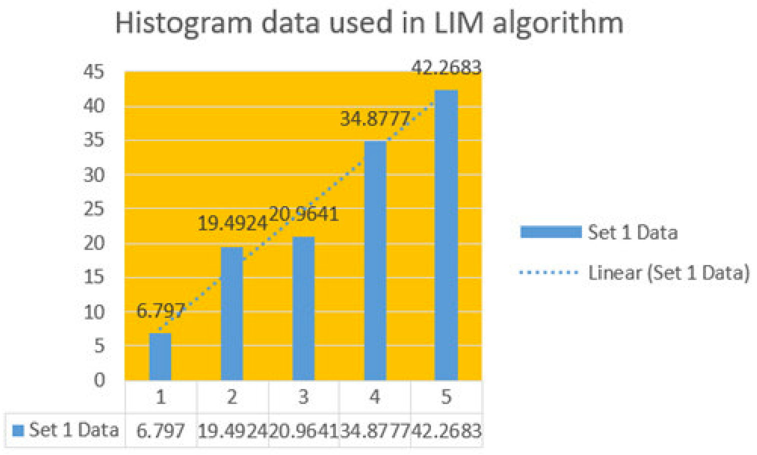


FIG. 4.8. *Output points of new key expansion (stage 8)*

TABLE 4.6
Output of stage 1, stage 2 and stage 3 (existing and new Key expansion)

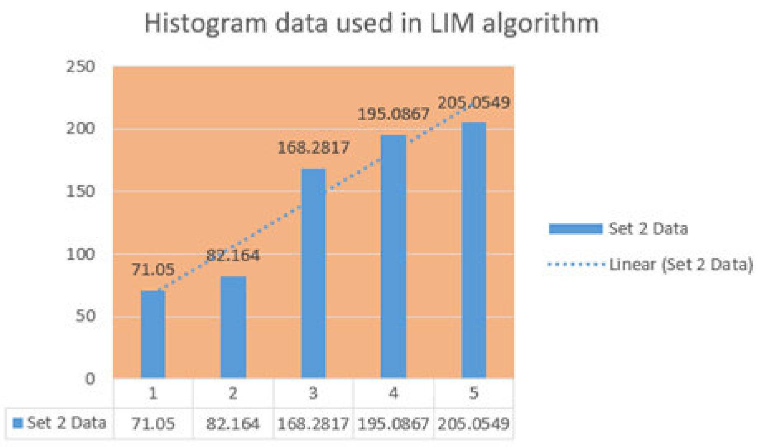
S.No.	Variable t11 (stage 1)	Output after Rotate_word ( ) (stage 2)	Output after Subtitute_word ( ) (stage 3)
1	09cf4f3c	cf4f3c09	8a84eb01
2	1f6d7251	6d72511f	3c40d1c0
3	9ea44a4e	a44a4e9e	49d62fb
4	5bef35eb	ef35eb5b	df96e939
5	fe58a041	58a041fe	6ae083bb
6	f088b5ae	88b5aef0	c4d5e48c
7	4371ac92	71ac9243	a3914f1a
8	a47a7ab5	7a7ab5a4	dadad549
9	8187771c81	87771c81	17f5acc
10	547d5bf2	7d5bf254	ff398920

As given in Table 4.7, Step 1 denotes Histogram Data (step 1), Step 3 denotes Sorted Data computed in the Algorithm (step 3), and Cval(LIM) denotes Computed value of LIM. The values of LIM for data used in extended Kummer surface 2 are 0.3451010 and 0.477174 for two different sample data points. Same values of LIM is obtained for surface 3.



	1	2	3	4	5
Set 1 Data	6.797	19.4924	20.9641	34.8777	42.2683

FIG. 4.9. *Histogram data used in LIM algorithm (surface 1) (set 1 data)*



	1	2	3	4	5
Set 2 Data	71.05	82.164	168.2817	195.0867	205.0549

FIG. 4.10. *Histogram data used in LIM algorithm (surface 1) (set 2 data)*

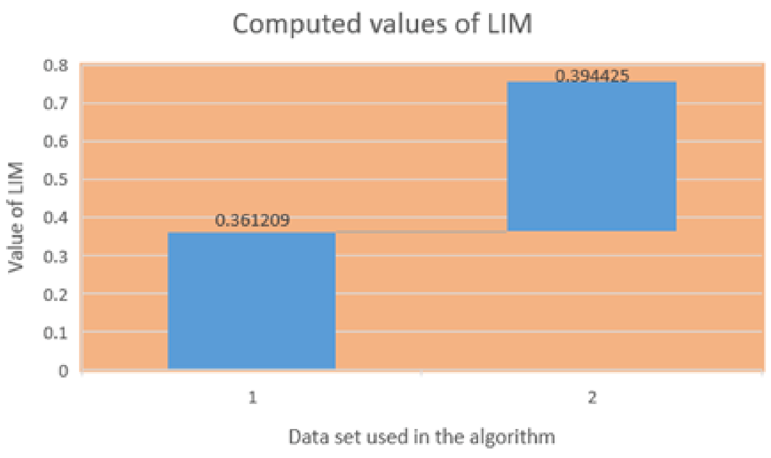


FIG. 4.11. *Computed values of LIM (surface 1)*

Table 4.7
Output of LIM (surface 1)

S. No.	Step 1	Step 3	Cval(LIM)
1	6.797	16.080608	0.361209
	19.4924	46.115883	
	20.9641	49.597691	
	34.8777	82.51503	
	42.2683	100	
2	71.05	34.649258	0.394425
	82.164	40.069271	
	168.2817	82.06665	
	195.0867	95.138763	
	205.0549	100	

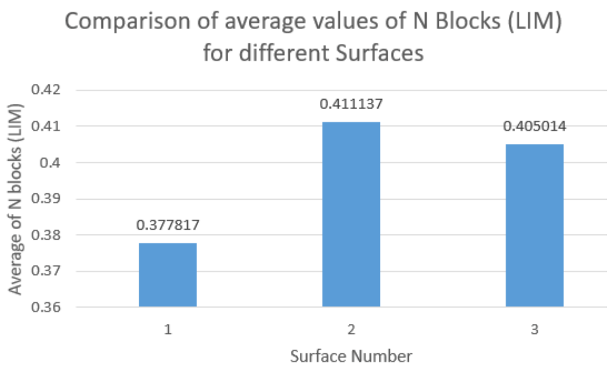


Fig. 4.12. *Comparison of average values of N Blocks (LIM) for different surfaces)*

4.3.1. Importance of computed values of LIM. An encrypting method, which has randomly distributed cipher texts inside the group of text used while encrypting, would be classified as a potential nice method encrypting. A computed value of LIM, that is, close to a value of 0.5 is an indication of randomly distributed [10]. Moreover, a drift from this value, significance occurrence of a kind of pattern in the cipher text [10]. The average values shown in Fig. 4.12 signifies that all the three surfaces are different, and there is a possibility of occurrence of a pattern, because these values drift from the value of 0.5.

5. Conclusion. This paper has a secure structure which contains mainly two aspects of a cryptographic system. Firstly, a scheme comprising of newly added point (u_5) in the kummer surface and then how this information inclusion could be utilized for a machine-learning algorithm in selecting an appropriate surface. Here, the selection of a particular surface based on a machine learning technique is provided. The work based on kummer surfaces and building a Fast Diffie-Hellman protocol is previously given in earlier papers. The focus of the proposed scheme is to enhance the security of the above-mentioned work by selecting an additional point (u_5) on the surface. Later, selection of multiple surfaces using NSGA II algorithm is given. Secondly, an encryption based on AES is taken. A new scheme for generating modified key-expansion in AES algorithm is described. The scheme builds a content-matrix using frequencies of different categories in the input message. Next, optimized values of the entries are chosen by running classical Lagrangian multiplier method. The selected entries are utilized in the modified key-expansion algorithm. Lastly, a brief overview of LIM index is given.

Acknowledgement: This publication is an outcome of the R & D work under taken project under the Visvesarvaraya PhD Scheme of Ministry of Electronics and IT, Government of India, being implemented by the Digital India Corporation.

REFERENCES

[1] N. P. Smart, and S. Siksek, *A Fast Diffie Hellman Protocol in Genus 2*, in Journal of cryptology, vol.12(1), 1999, pp. 67-73.
[2] H. Nover, *Algebraic cryptanalysis of AES: an overview*, in University of Wisconsin, USA, (2005).
[3] L. Ning, L.Kanfeng, L. Wenliang, and D. Zhongliang, *A joint encryption and error correction method used in satellite communications*, in China communications, vol. 11(3), (2014),pp. 70-79.
[4] Y. Dodis, R. Leonid, and S. Adam, *Fuzzy extractors: How to generate strong keys from biometrics and other noisy data*, in International conference on the theory and applications of cryptographic techniques, (2004), pp. 523-540.
[5] C. Peikert, and W. Brent, *Lossy trapdoor functions and their applications*, in SIAM Journal on Computing, vol.40(6), (2011), pp.1803-1844.
[6] A. C. Yao, *Theory and application of trapdoor functions*, in 23rd Annual Symposium on Foundations of Computer Science SFCS'08, (1982), pp.80-91.
[7] L. Blum, B. Manuel, and S. Mike, *A simple unpredictable pseudo-random number generator*, in SIAM Journal on computing, vol. 15(2), (1986), pp.364-383.
[8] P. S. Mehra, M. N. Doja, and B. Alam, *Codeword Authenticated Key Exchange (CAKE) light weight secure routing protocol for WSN*, International Journal of Communication Systems, 32 (2019).
[9] R. Cramer, and S. Victor, *A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack*, in Annual International Cryptology Conference, (1982), pp. 13-25.
[10] V. Karuvandan, C. Senthamarai and P. Shantharajah, in *Cryptanalysis of AES-128 and AES-256 block ciphers using lorenz information measure*, in Int. Arab J. Inf. Technology, vol. 13(6B), (2016), pp. 1054-1060.
[11] N. Ferguson, K. John, L. Stefan, S. Bruce, S. Mike, W.David, and W.Doug, *Improved cryptanalysis of Rijndael*, in International Workshop on Fast Software Encryption, 2000, pp. 213-230.
[12] A. Kaminsky, K. Michael, and R. Stanisaw, *An overview of cryptanalysis research for the advanced encryption standard*, in MILITARY COMMUNICATIONS CONFERENCE, 2010-MILCOM, (2010), pp.1310-1316.
[13] S. Chen, L. Rodolphe, L. Jooyoung, S. Yannick, and S. John, *Minimizing the two-round EvenMansour cipher*, in Journal of Cryptology, vol. 31(4), (2018), pp. 1064-1119.
[14] M. Masoumi, and M. Hadi Rezayati, *Novel approach to protect advanced encryption standard algorithm implementation against differential electromagnetic and power analysis*, in IEEE Transactions on Information Forensics and Security, Vol. 10(2), 2015, pp.256-265.
[15] F. Courbon, JJ. A. Fournier, P. L. Moundi, and A. Tria, *Combining image processing and laser fault injections for characterizing a hardware AES*, in IEEE transactions on computer-aided design of integrated circuits and systems, vol. 34(6), (2015), pp.928-936.
[16] M. Taha, and P. Schaumont, *Key updating for leakage resiliency with application to AES modes of operation*, in IEEE transactions on information forensics and security, vol. 10(3), (2015), pp.519-528.
[17] C. H. Baek, J.H.Cheon, and H.Hong, *White-box AES implementation revisited*, in Journal of Communications and Networks, vol. 18(3), (2016), pp.273-287.
[18] R. L. Rivest, *Cryptography and machine learning*, in International Conference on the Theory and Application of Cryptology, (1991), pp.427-439.
[19] R. Bost, R. A. Popa, S. Tu, and S. Goldwasser, *Machine learning classification over encrypted data*, in NDSS, (2015).
[20] T. Graepel, L. Kristin, and M. Naehrig, *ML confidential: Machine learning on encrypted data*, in International Conference on Information Security and Cryptology,(2012), pp. 1-21.
[21] F. M. Barbosa, A.R.S.F. Vidal, H.L.S. Almeida, and F.L. de Mello, *Machine Learning Applied to the Recognition of Cryptographic Algorithms Used for Multimedia Encryption*, in IEEE Latin America Transactions, vol. 15(7),(2017), pp. 1301-1305.
[22] H. Maghrebi, T.Portigliatti, and E. Prouff, *Breaking cryptographic implementations using deep learning techniques*, in International Conference on Security, Privacy, and Applied Cryptography Engineering, (2016), pp. 3-26.
[23] J.S. Muller, *Explicit Kummer surface formulas for arbitrary characteristic*, in LMS Journal of Computation and Mathematics, vol. 13, (2010), pp. 47-64.
[24] A. Garbagnati, and A. Sarti, *Kummer surfaces and K3 surfaces with* $(\frac{Z}{2Z})^4$ *symplectic action*, in Rocky Mountain Journal of Mathematics, vol. 46(4),(2016), pp. 1141-1205.
[25] Standard, Advance Encryption, *Federal information processing standards publication 197*, in FIPS PUB, (2001), pp. 46-53.
[26] C. Kumar, and M.N.Doja, *A Novel Framework for Portfolio Selection Model Using Modified ANFIS and Fuzzy Sets*, in Computers, vol. 7(4), (2018), pp. 57-64.
[27] R. K. Chahar, G. Datta, and N. Rajpal, *Design of a new Security Protocol*, in International Conference on Computational Intelligence and Multimedia Applications, vol. 4, (2007), pp. 132-136.
[28] N. Ferguson, K. John, L. Stefan, S. Bruce, S. Mike, W. David, and W. Doug, *Improved cryptanalysis of Rijndael*, in International Workshop on Fast Software Encryption, (2000), pp. 213-230.
[29] K. Deb, A.Pratap, S. Agarwal, and T. A. M. T. Meyarivan, *A fast and elitist multiobjective genetic algorithm: NSGA-II*, in IEEE transactions on evolutionary computation, vol. 6(2), (2002), pp. 182-197.

Edited by: Khaleel Ahmad
Received: Nov 25, 2018
Accepted: Feb 11, 2019