EUROPEAN JOURNAL OF PURE AND APPLIED MATHEMATICS
Vol. 17, No. 2, 2024, 979-995
ISSN 1307-5543 – ejpam.com
Published by New York Business Global

A Modified AES-512 Bits Algorithm for Data Encryption

Kwame Owusu Bempah[1,*], Kwasi Baah Gyamfi[2], Francis Ohene Boateng[3], Isaac Owusu-Mensah[4]

[1] *Department of Computer Science and Information Technology, Faculty of Health and Applied Science, Christian Service University College, Kumasi, Ghana*
[2] *Department of Mathematics, Faculty of Physical Science , Kwame Nkrumah University of Science and Technology, Kumasi, Ghana*
[3] *Department of Mathematics Education, Faculty of Applied Science and Mathematics Education, Akenten Appiah- Menka University of Skills Training and Entrepreneurial Development, Kumasi, Ghana*
[4] *Department of Integrated Science Education, Faculty of Science Education, Akenten Appiah-Menka University of Skills Training and Entrepreneurial Development, Kumasi, Ghana*

Abstract. Privacy is given top priority as far as communication involving transfer of confidential document or data is concerned. There is therefore a need to keep confidential data secretive from being invaded by unauthorised access, and this is done through cryptography algorithms, where the Advanced Encryption Standard algorithm has been the widely accepted symmetric block encryption algorithm for such purposes. This paper therefore focuses on developing a new AES-512 bits symmetric encryption algorithm through modification of the conventional AES-128 algorithm to be used purposefully in the classroom for document transfer.The development comes with increasing the plaintext bits of the conventional AES-128 algorithm to 512 bits plaintext which undergoes five operational transformations: STATE, SKGF, SRL, SCL and AARC through key size in the Galois field, $GF(2^9)$. A numerical example is then given to explain the use of the algorithm, and finally, we provide a comparative study of this algorithm and other existing symmetric encryption models, such as the AES-128 and DES algorithms.

2020 Mathematics Subject Classifications: 68P25, 68P27, 68P30

Key Words and Phrases: Cryptography, Encryption, Symmetric Algorithm, AES

*Corresponding author.
DOI: https://doi.org/10.29020/nybg.ejpam.v17i2.5114

Email addresses: kowusubempah@csuc.edu.gh (K.Owusu Bempah), kbgyamfi.cos@knust.edu.gh (K.Baah Gyamfi), foboateng@aamusted.edu.gh (F.O. Boateng) iowusumensah@aamusted.edu.gh (I.Owusu- Mensah)

1. Introduction

Cryptography involves studying and using mathematical techniques called algorithms to transform crucial messages, data or information to unclear or unreadable format [4] and comprises of encryption and decryption. Encryption simply means converting or turning a clear data, document or message into an unreadable format called ciphertext using algorithms or techniques whiles decryption is the process of converting the unclear message into a clear readable format called plaintext.

In this era, cryptography branches into mathematics and computer science and has an alliance with information theory and security, which play a vital role in security in computing, communication in mobile phones, passwords in computing, and even engineering, unlike the ancient time when cryptography consisted of only the encryption and decryption of messages using keys. So many applications resulted to a technique called Data Encryption Standard(DES) which is now becoming out of date due to the very small key size of 56-bits [2]. Due to the slowness of the DES algorithm implementation, the National Institute of Standards and Technology (NIST) introduced the Advanced Encryption Standard (AES) with the ultimate aim of being both faster and also protected. The Advanced Encryption Standard, AES, uses plaintext block lengths of 128, 192, and 256 bits with either a key size of 128, 192, or 256 bits, respectively, through 10 rounds, 12 rounds, and 14 rounds of encryption respectively. Currently, the advanced encryption standard algorithm has been expanded and enhanced through development due to the numerous AES attackers. In view of this, researchers are still persisting in developing algorithms that would be able to deter intruders from invading confidential documents or data.

A high speed and highly restricted image encryption algorithm in [9] was proposed by modifying the AES algorithm to improve its performance by decreasing both the computation cost and hardware requirement and also enhancing the security level. The only drawback in the process was that the encryption and decryption times really increased, and the attacks on the proposed algorithm could also lead to a reduction in the number of rounds of the algorithm.

The research work of [5], discovered an efficient method for implementing the function of the AES byte substitution (S-box). This was focused on implementing the AES in non-volatile FPGAs. It was found that the proposed method uses minimum space and runs much faster than the one that requires using the whole S-box in the logic area, but the only drawback was that FPGAs cannot function in areas with low battery cases.

In the research paper [8], an efficient implementation of the AES algorithm was adopted on an FPGA using the VirtexE family of devices (XCV812), where the results of the sequential pipeline architecture and that of pipeline architecture were compared. It was found that, the design in using sequential took 2744 CLB with a throughput of 258.5M, while 2136 CLB slices with throughput of 2868Mbits were obtained for the pipeline. Using the pipeline for design can be more efficient based on area and throughput. But there could be comparison using various techniques such as parallelization, loop unrolling, or memory for implementing the AES algorithm to determine the effect on area and throughput as well.

In [6], a new version of the conventional AES algorithm was introduced using an input block of 512 bits with a key size of 512 bits and resources such as processor and memory. The parameters of the AES 128 and 256 bits were compared to the new version and showed that the new version produced greater confidentiality and throughput and was resistant to linear and differential attacks. In the process, an increment of 230% in the throughput was realised compared to the AES-128 algorithm. The only drawback in the process was that there should be a performance analysis of the proposed algorithms and other symmetric algorithms to evaluate their security and other resources. In [7], an algorithm for advanced encryption standard was proposed, where sub-keys were produced differently from the main key and each sub-key is used to encrypt the AES round singly, but the proposed algorithm was very slow compared to the conventional AES algorithm and was resistant to brute force attack.
Further, in [1], a 4D teserract symmetric block was redesigned to be of good territory for the manufacture of encryption keys, and four rounds and other sub rounds are engaged to shift the teserract to be able to randomly produce some of the encryption keys. The XOR operation technique is used in this method due to its speed and lightweight. A T-0,1 512 key was compared to the AES-256 to evaluate their speed performance, and the results show that the assumed key size was accepted based on the International Telecommunication Union (ITU-T).
All the algorithms presented above are used in industrial and multimedia settings, which involve networking and require the use of the internet to transfer or store confidential data. There is no algorithm for AES that has been developed practically for educational purposes. Motivated by this, in this paper, we develop a new AES-512 bits symmetric encryption algorithm through modification of the conventional AES-128 algorithm using five operational transformations: STATE, SKGF SRL, SCL, and AARC, to be used purposefully in the classroom for document transfer.

2. Preliminaries

This section discusses the generation of the proposed AES-512-bit algorithm keys in Galois Field, $GF(2^9)$, which are given in bit strings.

2.1. Key generation process for the proposed AES-512 bits algorithm in $GF(2^r)$

Every degree in the Galois field, $GF(2^r)$,is generally represented by the polynomial $a_{r-1}x^{r-1} + a_{r-2}x^{r-2} + a_{r-3}x^{r-3} + ... + a_{r-r}$, where r is the degree in the field. This implies that in $GF(2^9)$, the corresponding polynomial with $r = 9$ is given as;
$a_8x^8 + a_7x^7 + a_6x^6 + a_5x^5 + a_4x^4 + a_3x^3 + a_2x^2 + a_1x + a_0$ Where a_0, a_1,to a_8 represent the coefficients of the polynomial and serve as bit strings.The bit strings to be used for the polynomial in our proposed algorithm to represent the keys are gotten in $GF(2^9)$.

2.2. Proposed approach for generation of Bit strings in $GF(p^r)$

For any Galois field, $GF(p^r)$, where r= degree and p=2, the powers or exponents in the field are given as: $1 \geq x \geq r$ or $1 \leq x \leq r$.

1 In our modified algorithm, $GF(p^9)$, $r = 9$ implies that the powers of the exponent are given as:
$\{1 \geq x \geq 9\}$ implying $x = 9, 8, 7, 6, 5, 4, 3, 2, 1$

2 The values of x are raised to the powers of p to get the Total Number of Strings (TNS) for each value of x as, TNS= 2^x

3 The estimated Total Number of Strings (TNS) needs to have a pattern as to how the strings are written. This is given as: $PTNS = \frac{TNS}{2}$. The division by 2 is a result of bits, which are in binary mode. Where PTNS is Pattern for Total Number of Strings.

Example 1: For $x = 9$ as a power in the Galois field, then $TNS = 2^x = 2^9 = 512$ and $PTNS = \frac{TNS}{2} = 256$. This means we produce 256 (ones) continuously, followed by 256 (zeros), or vice versa.

Example 2: Also, for $x = 8$ as a power in the Galois field, then $TNS = 2^x = 2^8 = 256$ and $PTNS = \frac{TNS}{2} = 128$. This means we produce 128 (ones) continuously, followed by 128 (zeros), or vice versa. The process is repeated again to make up the TNS.
The following strings are produced for each value of x, which is substituted into the polynomial in $GF(2^9)$, and the results for the polynomial have been arranged according to its degree.
000000000
000000001
degree 1= 000000010, 000000011
degree 2= 000000100, 000000101, 000000110, 000000111
degree 3= 000001000, 000001001, 000001010, 000001011, 000001100, 000001101, 000001110, 000001111
degree 4= ,000010000, 000010001, 000010010, 000010011, 000010100, 000010101, 000010110, 000010111, 000011000, 000011001, 000011010, 000011011, 000011100, 000011101, 000011110, 000011111
degree 5= 000100000,000100001,000100010,000100011, 000100100,000100101, ,000100110, 00100111,000101000,000101001,000101010,000101011,000101100, 000101101, 000101110,000101111,000110000,000110001,000110010,000110011, 000110100,000110101,000110110,000110111,000111000,000111001, 000111010,000111011,000111100,000111101,000111110, 000111111
degree 6= 001000000,001000001,001000010,001000011,001000100,001000101, 001000110,001000111,001001000,001001001,001001010,001001011,001001100, 001001101,001001110,001001111,001010000,001010001,001010010,001010011, 001010100,001010101,001010110,001010111,001011000,001011001,001011010, 001011011,001011100,001011101,001011110,001011111,001100000,001100001,

001100010,001100011,001100100,001100101,001100110,001100111,001101000,
001101001,001101010,001101011,001101100,001101101,001101110,001101111,
001110000,001110001,001110010,001110011,001110100,001110101,001110110,
001110111,001111000,001111001,001111010,001111011,001111100,001111101,
001111110,001111111
degree 7= 010000000,010000001,010000010,010000011,010000100,010000101,
010000110,010000111,010001000,010001001,010001010,010001011,010001100,
010001101,010001110,010001111,010010000,010010001,010010010,010010011,
010010100,010010101,010010110,010010111,010011000,010011001,010011010,
010011011,010011100,010011101,010011110,010011111,010100000,010100001,
010100010,010100011,010100100,010100101,010100110,010100111,010101000,
010101001,010101010,010101011,010101100,010101101,010101110,010101111,
010110000,010110001,010110010,010110011,010110100,010110101,010110110,
010110111,010111000,010111001,010111010,010111011,010111100,010111101,
010111110,010111111,011000000,011000001,011000010,011000011,011000100,
011000101,011000110,011000111,011001000,011001001,011001010,011001011,
011001100,011001101,011001110,011001111,011010000,011010001,011010010,
011010011,011010100,011010101,011010110,011010111,011011000,011011001,
011011010,011011011,011011100,011011101,011011110,011011111,011100000,
011100001,011100010,011100011,011100100,011100101,011100110,011100111,
011101000,011101001,011101010,011101011,011101100,011101101,011101110,
011101111,011110000,011110001,011110010,011110011,011110100,011110101,
011110110,011110111,011111000,011111001,011111010,011111011,011111100,
011111101,011111110,011111111
degree 8= 100000000,100000001,100000010,100000011,100000100,100000101,
100000110,100000111,100001000,100001001,100001010,100001011,100001100,
100001101,100001110,100001111,100010000,100010001,100010010,100010011,
100010100,100010101,100010110,100010111,100011000,100011001,100011010,
100011011,100011100,100011101,100011110,100011111,100100000,100100001,
100100010,100100011,100100100,100100101,100100110,100100111,100101000,
100101001,100101010,100101011,100101100,100101101,100101110,100101111,
100110000,100110001,100110010,100110011,100110100,100110101,100110110,
100110111,100111000,100111001,100111010,100111011,100111100,100111101,
100111110,100111111,101000000,101000001,101000010,101000011,101000100,
101000101,101000110,101000111,101001000,101001001,101001010,101001011,
101001100,101001101,101001110,101001111,101010000,101010001,101010010,
101010011,101010100,101010101,101010110,101010111,101011000,101011001,
101011010,101011011,101011100,101011101,101011110,101011111,101100000,
101100001,101100010,101100011,101100100,101100101,101100110,101100111,
101101000,101101001,101101010,101101011,101101100,101101101,101101110,
101101111,101110000,101110001,101110010,101110011,101110100,101110101,
101110110,101110111,101111000,101111001,101111010,101111011,101111100,
101111101,101111110,101111111,110000000,110000001,110000010,110000011,

110000100,110000101,110000110,110000111,110001000,110001001,110001010,
110001011,110001100,110001101,110001110,110001111,110010000,110010001,
110010010,110010011,110010100,110010101,110010110,110010111,110011000,
110011001,110011010,110011011,110011100,110011101,110011110,110011111,
110100000,110100001,110100010,110100011,110100100,110100101,110100110,
110100111,110101000,110101001,110101010,110101011,110101100,110101101,
110101110,110101111,110110000,110110001,110110010,110110011,110110100,
110110101,110110110,110110111,110111000,110111001,110111010,110111011,
110111100,110111101,110111110,110000000,111000000,111000001,111000010,
111000011,111000100,111000101,111000110,111000111,111001000,111001001,
111001010,111001011,111001100,111001101,111001110,111001111,111010000,
111010001,111010010,111010011,111010100,111010101,111010110,111010111,
111011000,111011001,111011010,111011011,111011100,111011101,111011110,
111011111,111100000,111100001,111100010,111100011,111100100,111100101,
111100110,111100111,111101000,111101001,111101010,111101011,111101100,
111101101,
111101110,111101111,111110000,111110001,111110010,111110011,
111110100,111110101,111110110,111110111,111111000,111111001,111111010,
111111011,111111100,111111101,111111110,111111111

3. Methodology

This section describes the proposed approach for the modified AES-512 bit algorithm for the encryption of confidential documents or data to be used purposefully in the classroom. The modified AES-512 bit encryption algorithm uses plaintext of sixty-four bytes composed of a block array of 8×8 square matrix with a key size in Galois field, $GF(2^9)$. The plaintext is encrypted with a key using the algorithm,$A_i = V + M_i$
Where V is plaintext comprising of sixty-four characters made up of a block array of sixty-four bytes put in a (8×8) square block matrix. M = keys in $GF(2^9)$ and i= key initiation in the range $1 \leq i \leq 8$. The modified algorithmic encryption is accomplished through the following five operational transformations: STATE, SKGF, SRL, SCL, and AARC. The justification for these steps is given alongside.

STATE transformation: In this stage, the plaintext with characters of sixty-four byte is put in a (8×8) square block array of matrix. An initial key, say $i = 1$, containing sixty-four keys, is needed to initiate the algorithm, and this key is chosen randomly within $GF(2^9)$ and written in a (8×8) square matrix of block array. It is then executed using the algorithm, $A_1 = V + M_1$.
This step is purposely to put the plaintext into a square block array matrix, and by arranging the plaintext in this pattern, it accelerates the ensuing encryption operations and ensures the processing of data adequately and completely.

SKGF transformation: This stage is called Sub Keys from Galois Field (SKGF), and

the sub keys within the field, $GF(2^9)$ for $i = 2, 3, 4, 5, 6, 7, 8$, represent the sub keys, which comprise of sixty-four keys each randomly chosen and initiated after the initiation of $i = 1$ to encrypt the same plaintext, V producing the sub algorithm, $A_i = \sum_{i=2}^{8}(V + M_i)$. This means the plaintext in this stage undergoes again seven series of encryption processes.
This transformation is intended to produce round keys after the initial key, and producing the round keys is crucial for the AES encryption technique as it ensures the complexity of the algorithmic encryption, which promotes and enhances confidentiality.

SRL transformation: This stage is called Shift Row Left Transformation (SRL), where the subscript of A from the output of the preceding stage would serve as the pivotal position, which would represent the type of row and number of times the position would be shifted to the left outside the matrix for another matrix to be formed.
This step actually involves shifting rows in the square block array matrix and thereby assisting in mixing and scrambling the data to increase diffusion, which is essential for attaining great encryption.

SCL transformation: This stage is called the Shift Column Left (SCL) transformation, where we shift the entire column of the matrix of the ciphertext gotten from the SRL stage. The subscript of A from the output of the preceding stage serves as the type of column and also the number of times to be shifted to the left outside the matrix to create a different matrix. Meaning A_i for $i = 1, 2, 3, 4, 5, 6, 7, 8$ from the preceding stage represents the first, second, and up to eighth columns would be shifted respectively depending on the value of i to the left outside the matrix.
This step is mixing of column operation which additionally increases the diffusion and confusion attributes of the algorithmic encryption which becomes more resistant to attacks.

AARC transformation: This is the last stage in the algorithm called Add All Round Ciphertext (AARC). A bulky ciphertext, A_b, is produced where all the ciphertexts in the preceding stage are added to get a bulky ciphertext.
This step involves extra operations to complete the encryption technique, and this final step is crucial for finalising the encryption process and generating the ciphertext that is protected to be ready for dissemination, propagation, and storage.
Generally, each transformational step plays an essential role in the modified AES-512 bits algorithm, providing overall security and strength to the encryption technique.

Numerical Example Supppose we want to encrypt a confidential data composed of plaintext characters of sixty-four bytes; **Cryptography is the process of encryption and decryption of data**
The plaintext characters is put into an array of 8×8 square matrix block denoted as V to form the first stage called **STATE**

$$V = \begin{bmatrix} C & r & y & p & t & o & g & r \\ a & p & h & y & space & i & s & space \\ t & h & e & space & p & r & o & c \\ e & s & s & space & o & f & space & e \\ n & c & r & y & p & t & i & o \\ n & space & a & n & d & space & d & e \\ c & r & y & p & t & i & o & n \\ space & o & f & space & d & a & t & a \end{bmatrix}$$

The characters would then be converted into their decimal mode values using the American Standard Code for Information Interchange, ($ASCII$) table [3] below.

Dec	Chr	Dec	Chr	Dec	Chr	Dec	Chr	Dec	Chr	Dec	Chr
32	space	48	0	64	@	80	P	96	,	112	p
33	!	49	1	65	A	81	Q	97	a	113	q
34	”	50	2	66	B	82	R	98	b	114	r
35	#	51	3	67	C	83	S	99	c	115	s
36	$	52	4	68	D	84	T	100	d	116	t
37	%	53	5	69	E	85	U	101	e	117	u
38	&	54	6	70	F	86	V	102	f	118	v
39	’	55	7	71	G	87	W	103	g	119	w
40	(	56	8	72	H	88	X	104	h	120	x
41	)	57	9	73	I	89	Y	105	i	121	y
42	∗	58	:	74	J	90	Z	106	j	122	z
43	+	59	;	75	K	91	[	107	k	123	{
44	,	60	<	76	L	92	\	108	l	124	|
45	-	61	=	77	M	93	]	109	m	125	}
46	.	62	>	78	N	94	ˆ	110	n	126	˜
47	/	63	?	79	O	95	_	111	o		

This produces the numerical values for the plaintext V below.

$$V = \begin{bmatrix} 67 & 114 & 121 & 112 & 116 & 111 & 103 & 114 \\ 97 & 112 & 104 & 121 & 32 & 105 & 115 & 32 \\ 116 & 104 & 101 & 32 & 112 & 114 & 111 & 99 \\ 101 & 115 & 115 & 32 & 111 & 102 & 32 & 101 \\ 110 & 99 & 114 & 121 & 112 & 116 & 105 & 111 \\ 110 & 32 & 97 & 110 & 100 & 32 & 100 & 101 \\ 99 & 114 & 121 & 112 & 116 & 105 & 111 & 110 \\ 32 & 111 & 102 & 32 & 100 & 97 & 116 & 97 \end{bmatrix}$$

The plaintext V is substituted into our modified AES-512 bits algorithm, and an initial key, say $i = 1$, representing the first sixty-four keys, is chosen randomly from the Galois field, $GF(2^9)$. This produces the algorithm $A_1 = V + M_1$ which is given as

$$A_1 = \begin{bmatrix} 67 & 114 & 121 & 112 & 116 & 111 & 103 & 114 \\ 97 & 112 & 104 & 121 & 32 & 105 & 115 & 32 \\ 116 & 104 & 101 & 32 & 112 & 114 & 111 & 99 \\ 101 & 115 & 115 & 32 & 111 & 102 & 32 & 101 \\ 110 & 99 & 114 & 121 & 112 & 116 & 105 & 111 \\ 110 & 32 & 97 & 110 & 100 & 32 & 100 & 101 \\ 99 & 114 & 121 & 112 & 116 & 105 & 111 & 110 \\ 32 & 111 & 102 & 32 & 100 & 97 & 116 & 97 \end{bmatrix} + M_1$$

Since the keys are polynomials, the plaintext, V, is converted into the polynomial mode by first converting into binary mode, after which the binary numbers will serve as bit strings to be used as coefficients for the polynomial in $GF(2^9)$ given in the table below.

V	Binary	polynomial,$GF(2^9)$	V	binary	polynomial,$GF(2^9)$
67	1000011	$1+x^5+x^6$	116	1110100	
97	1100001	$1+x+x^6$	32	100000	
116	1110100	$1+x+x^2+x^4$	112	1110000	$1+x+x^2$
101	1100101	$1+x+x^4+x^6$	111	1110011	$1+x+x^2+x^5+x^6$
110	1101110	$1+x+x^3+x^4+x^5$	112	1110000	$1+x+x^2$
110	1101110	$1+x+x^3+x^4+x^5$	100	1100100	$1+x+x^4$
99	1100011	$1+x+x^5+x^6$	116	1110100	$1+x+x^2+x^4$
32	100000	1	100	1100100	$1+x+x^4$
114	1110010	$1+x+x^2+x^5$	111	1110011	$1+x+x^2+x^5+x^6$
112	1110000	$1+x+x^2$	105	1101001	$1+x+x^3+x^6$
104	1101000	$1+x+x^3$	114	1110010	$1+x+x^2+x^5$
115	1110011	$1+x+x^2+x^5+x^6$	102	1100110	$1+x+x^4+x^5$
99	1100011	$1+x+x^5+x^6$	116	1110100	$1+x+x^2+x^4$
32	100000	1	32	100000	1
114	1110010	$1+x+x^2+x^5$	105	1101001	$1+x+x^3+x^6$
111	1110011	$1+x+x^2+x^5+x^6$	97	1100001	$1+x+x^6$
121	1111001	$1+x+x^2+x^3+x^6$	103	1100111	$1+x+x^4+x^5+x^6$
104	1101000	$1+x+x^3$	115	1110011	$1+x+x^2+x^5+x^6$
101	1100101	$1+x+x^4+x^6$	111	1110011	$1+x+x^2+x^5+x^6$
115	1110011	$1+x+x^2+x^5+x^6$	32	100000	1
114	1110010	$1+x+x^2+x^5$	105	1101001	$1+x+x^3+x^6$
97	1100001	$1+x+x^6$	100	1100100	$1+x+x^4$
121	1111001	$1+x+x^2+x^3+x^6$	111	1110011	$1+x+x^2+x^5+x^6$
102	1100110	$1+x+x^4+x^5$	116	1110100	$1+x+x^2+x^4$
112	1110000	$1+x+x^2$	114	1110010	$1+x+x^2+x^5$
121	1111001	$1+x+x^2+x^3+x^6$	32	100000	1
32	100000	1	99	1100011	$1+x+x^5+x^6$
32	100000	1	101	1100101	$1+x+x^4+x6$
121	1111001	$1+x+x^2+x^3+x^6$	111	1110011	$1+x+x^2+x^5+x^6$
110	1101110	$1+x+x^3+x^4+x^5$	101	1100101	$1+x+x^4+x^6$
112	1110000	$1+x+x^2$	110	1101110	$1+x+x^3+x^4+x^5$
32	100000	1	97	1100001	$1+x+x^6$

$A_1 =$

$$\begin{bmatrix} 1+x^5+x^6 & 1+x+x^2+x^5 & 1+x+x^2+x^3+x^6 & 1+x+x^2 \\ 1+x+x^6 & 1+x+x^2 & 1+x+x^3 & 1+x+x^2+x^3+x^6 \\ 1+x+x^2+x^4 & 1+x+x^3 & 1+x+x^4+x^6 & 1 \\ 1+x+x^4+x^6 & 1+x+x^2+x^5+x^6 & 1+x+x^2+x^5+x^6 & 1 \\ 1+x+x^3+x^4+x^5 & 1+x+x^5+x^6 & 1+x+x^2+x^5 & 1+x+x^2+x^3+x^6 \\ 1+x+x^3+x^4+x^5 & 1 & 1+x+x^6 & 1+x+x^3+x^4+x^5 \\ 1+x+x^5+x^6 & 1+x+x^2+x^5 & 1+x+x^2+x^3+x^6 & 1+x+x^2 \\ 1 & 1+x+x^2+x^5+x^6 & 1+x+x^4+x^5 & 1 \end{bmatrix}$$

$$\begin{bmatrix} 1+x+x^2+x^4 & 1+x+x^2+x^5+x^6 & 1+x+x^4+x^5+x^6 \\ 1 & 1+x+x^3+x^6 & 1+x+x^2+x^5+x^6 \\ 1+x+x^2 & 1+x+x^2+x^5 & 1+x+x^2+x^5+x^6 \\ 1+x+x^2+x^5+x^6 & 1+x+x^4+x^5 & 1 \\ 1+x+x^2 & 1+x+x^2+x^4 & 1+x+x^3+x^6 \\ 1+x+x^4 & 1 & 1+x+x^4 \\ 1+x+x^2+x^4 & 1+x+x^3+x^6 & 1+x+x^2+x^5+x^6 \\ 1+x+x^4 & 1+x+x^6 & 1+x+x^2+x^4 \end{bmatrix}$$

$$\begin{matrix} 1+x+x^2+x^5 \\ 1 \\ 1+x+x^5+x^6 \\ 1+x+x^4+x^6 \\ 1+x+x^2+x^5+x^6 \\ 1+x+x^4+x^6 1+x+x^3+x^4+x^5 \\ 1+x+x^6 \end{matrix} \Bigg]$$

$+ M_1$

This produces ciphertext in polynomials for encryption round 1. We now proceed to the **SKGF** transformation, where the sub keys from the $GF(2^9)$ produce the sub-algorithms for $i = 2$ to 8. Thus

$$A_i = \sum_{i=2}^{8} (V + M_i)$$

produces;

$$\begin{bmatrix} A_{i_{(11)}} & \cdots & A_{i_{(18)}} \\ \vdots & \ddots & \vdots \\ A_{i_{(81)}} & \cdots & A_{i_{(88)}} \end{bmatrix} = \begin{bmatrix} V_{(11)} & \cdots & V_{(18)} \\ \vdots & \ddots & \vdots \\ V_{(81)} & \cdots & V_{(88)} \end{bmatrix} + \begin{bmatrix} M_{i_{(11)}} & \cdots & M_{i_{(18)}} \\ \vdots & \ddots & \vdots \\ M_{i_{(81)}} & \cdots & M_{i_{(88)}} \end{bmatrix}$$

This means the same plaintext, V, undergoes seven series of encryption processes with

different keys chosen randomly from $GF(2^9)$ up to encryption round 8.We then embark on the next stage for encryption round 9, which is the **SRL** transformation, where the output from the STATE and SKGF becomes the input, with subscripts i and j representing the matrix of the i^{th} row and j^{th} column, respectively, and $1 \leq R \leq 8$ represents $Row1$ to $Row8$

Algorithm 1 SRL Transformation

$$A_{i_{(SRL)}} = \sum_{i=1}^{8} \sum_{j=k+1}^{q} A_{ij}(V + M_i)$$

Ensure: $R1 \longleftarrow A_{1_{(STATE)}}$
if $i = 1$ for $q = 8$ **then**
    $k = 0$
else
    if $i = 1$ for $q = 1$ **then**
        $k = 0$
Ensure: $R2 \longleftarrow A_{2_{(SKGF)}}$
        if $i = 2$ for $q = 8$ **then**
            $k = 2$
        else
            if $i = 2$ for $q = 2$ **then**
                $k = 0$
Ensure: $R3 \longleftarrow A_{3_{(SKGF)}}$
                if $i = 3$ for $q = 8$ **then**
                    $k = 3$
                else
                    if $i = 3$ for $q = 3$ **then**
                        $k = 0$
Ensure: $R4 \longleftarrow A_{4_{(SKGF)}}$
                        if $i = 4$ for $q = 8$ **then**
                            $k = 4$
                        else
                            if $i = 4$ for $q = 4$ **then**
                                $k = 0$

Algorithm 2 Continuation

Ensure: $R5 \longleftarrow A_{5_{(SKGF)}}$
if $i = 5$ for $q = 8$ **then**
    $k = 5$
else
    if $i = 5$ for $q = 5$ **then**
        $k = 0$
Ensure: $R6 \longleftarrow A_{6_{(SKGF)}}$
        if $i = 6$ for $q = 8$ **then**
            $k = 6$
        else
            if $i = 6$ for $q = 6$ **then**
                $k = 0$
Ensure: $R7 \longleftarrow A_{7_{(SKGF)}}$
                if $i = 7$ for $q = 8$ **then**
                    $k = 7$
                else
                    if $i = 7$ for $q = 7$ **then**
                        $k = 0$
Ensure: $R8 \longleftarrow A_{8_{(SKGF)}}$
                        if $i = 8$ for $q = 8$ **then**
                            $k = 0$

We then proceed to the **SCL** transformation and the last stage, which is the **AARC** transformation for encryption rounds 9 and 10, respectively, where the output from the SRL stage becomes the input for the SCL transformation. This produces a bulky ciphertext, A_b, for encryption round 11. The algorithm below is used to realise the ciphertext.

Algorithm 3 SCL Transformation: Return $A_{ij_{(SRL)}}$ from Step8

$$A_{i_{(SCL)}} = \sum_{i=1}^{8} \sum_{j=k+1}^{q} A_{ij_{(SRL)}}(V + M_i)$$

Ensure: $C1 \longleftarrow A_{1_{(SRL)}}$
if $i = 1 \; \forall \; i \neq j$ for $1 \leq i \leq 8$ **then**
  $j = k + 1$ for $k = 1$
else
  if $2 \leq i \leq 8$ **then**
    $j = k + 1$ for $k = 1$
Ensure: $C2 \longleftarrow A_{2_{(SRL)}}$
    if $1 \leq 1 \leq 2 \; \forall \; i \neq j$ for $1 \leq i \leq 8$ **then**
      $j = 2k + 1$ for $1 \leq k = 2$
    else
      if $3 \leq i \leq 8$ **then**
        $j = 2k + 1$ for $k = 1$
Ensure: $C3 \longleftarrow A_{3_{(SRL)}}$
        if $1 \leq i \leq 2 \; \forall \; i \neq j$ **then**
          $j = 3k + 1$ for $k = 1$
        else
          if $i = 3$ **then**
            $j = 3k + 1$ for $k = 2$
          else
            if $4 \leq i \leq 8$ **then**
              $j = 3k + 1$ for $k = 1$
Ensure: $C4 \longleftarrow A_{4_{(SRL)}}$
              if $1 \leq i \leq 3 \; \forall \; i \neq j$ **then**
                $j = 4k + 1$ for $k = 1$
              else
                if $i = 4$ **then**
                  $j = 4k + 1$ for $k = 0$
                else
                  if $5 \leq i \leq 8$ **then**
                    $j = 4k + 1$ for $k = 1$

Algorithm 4 Continuation SCL Transformation

Ensure: $C5 \longleftarrow A_{5_{(SRL)}}$
if $1 \leq i \leq 4 \; \forall \; i \neq j$ **then**
  $j = 5k + 1$ for $k = 1$
else
  if $i = 5$ **then**
    $j = 3$
  else
    if $6 \leq i \leq 8$ **then**
      $j = 5k + 1$ for $k = 1$
Ensure: $C6 \longleftarrow A_{6_{(SRL)}}$
      if $1 \leq i \leq 5 \; \forall \; i \neq j$ **then**
        $j = 6k + 1$ for $k = 1$
      else
        if $i = 6$ **then**
          $j = 5$
        else
          if $7 \leq i \leq 8$ **then**
            $j = 6k + 1$ for $k = 1$
Ensure: $C7 \longleftarrow A_{7_{(SRL)}}$
            if $1 \leq i \leq 6 \; \forall \; i \neq j$ **then**
              $j = 7k + 1$ for $k = 1$
            else
              if $i = 7$ **then**
                $j = 7$
              else
                if $i = 8$ **then**
                  $j = 7k + 1$ for $k = 1$
Ensure: $C8 \longleftarrow A_{8_{(SRL)}}$
                  if $1 \leq i \leq 8$ **then**
                    $1 \leq j \leq 8$
AARC transformation
return $A_{ij_{(SCL)}}$ for $1 \leq i \leq 8$ from step36 to step39

$$A_b = \sum_{i=1}^{8} A_{ij_{(SCL)}}$$

Following the proceedings of this algorithm, we provide a comparative study of the modified algorithm and existing symmetric encryption algorithms like AES-128 and DES. **Key Size:** The modified AES-512 algorithm uses a key size of 512 bits, which provides better security and makes it resistant to different types of attacks, like differential and linear attacks, whereas the DES algorithm and AES-128 algorithm use a key size of 56

bits and 128 bits, respectively. But the DES algorithm becomes vulnerable to attacks such as brute force attacks due to its smaller key size, whereas the AES-128 algorithm is less prone to attacks.

Block Size: The DES algorithm and the AES-128 algorithm operate with a block size of 64 bits and 128 bits, respectively, while the modified AES-512 bits algorithm uses a block size of 64 bits to process data. The large block size in the modified algorithm enables more data to be encrypted in each block, which could lead to improved efficiency.

Encryption Rounds: The DES algorithm and the AES-128 algorithm undergo 16 rounds of encryption and 10 rounds of encryption, respectively, whereas the modified AES-512 bits algorithm requires a different number of rounds to cater for the larger key size and block size. The modified algorithm's additional rounds may enhance security but could slow down during execution.

Security: The key size of the DES algorithm has been broken and becomes vulnerable to attacks due to its smaller size. Both the AES-128 and AES-512 algorithms are based on the advanced encryption standard (AES), but the modified AES-512 algorithm provides improved security features and offers resistance against several forms of attack, such as linear and differential attacks, making it resilient to protect confidential documents.

Adaptability: The modified AES-512 bits algorithm was developed specifically for educational purposes for document transfer, and the AES-128 bits algorithm is an established encryption algorithm used in the industry for data security and communication networks. The DES algorithm is not able to meet modern application requirements for security due to restrictions in key size and encryption strength.

4. Conclusion

Due to the need to always secure confidential documents, a modified AES-512 bits algorithmic encryption approach has been developed specifically to be used in the classroom for the encryption of confidential documents or data. The algorithm produces keys for encryption of the data in polynomial mode, making it difficult for intruders to invade the data. The algorithm could be extended in the future to provide performance analysis of the modified algorithm with other symmetric encryption algorithms to provide insights into the efficiency and effectiveness of the modified algorithm and its potential for real-world applications.

Acknowledgements

The authors thank the reviewers for their contributions. We acknowledge that the inputs offered by the editor and reviewers have really helped us improve the paper.

References

[1] Ali M Alshahrani. Develp a 512–bit high speed symmertic algorithm. In *2016 Universal Technology Management Conference (UTMC)*, page 62, 2016.

[2] Ross Anderson, Eli Biham, and Lars Knudsen. Serpent: A proposal for the advanced encryption standard. *NIST AES Proposal*, 174:1–23, 1998.

[3] Christoforus Juan Benvenuto. Galois field in cryptography. *University of Washington*, 1(1):1–11, 2012.

[4] Albrecht Beutelspacher. *Cryptology*. MAA, 1994.

[5] Lubos Gaspar, Milos Drutarovsky, Viktor Fischer, and Nathalie Bochard. Efficient aes s-boxes implementation for non-volatile fpgas. In *2009 International Conference on Field Programmable Logic and Applications*, pages 649–653. IEEE, 2009.

[6] Rishabh Jain, Rahul Jejurkar, Shrikrishna Chopade, Someshwar Vaidya, and Mahesh Sanap. Aes algorithm using 512 bit key implementation for secure communication. *International Journal of Advanced Research in Computer Science*, 1(3), 2014.

[7] Shaaban Sahmoud, Wisam Elmasry, and Shadi Abudalfa. Enhancement the security of aes against modern attacks by using variable key block cipher. *Int. Arab. J. e Technol.*, 3(1):17–26, 2013.

[8] Nazar Abbas Saqib, Francisco Rodríguez-Henríquez, and Arturo Diaz-Perez. Aes algorithm implementation-an efficient approach for sequential and pipeline architectures. In *Proceedings of the Fourth Mexican International Conference on Computer Science, 2003. ENC 2003.*, pages 126–130. IEEE, 2003.

[9] Salim Muhsin Wadi and Nasharuddin Zainal. High definition image encryption algorithm based on aes modification. *Wireless personal communications*, 79(2):811–829, 2014.